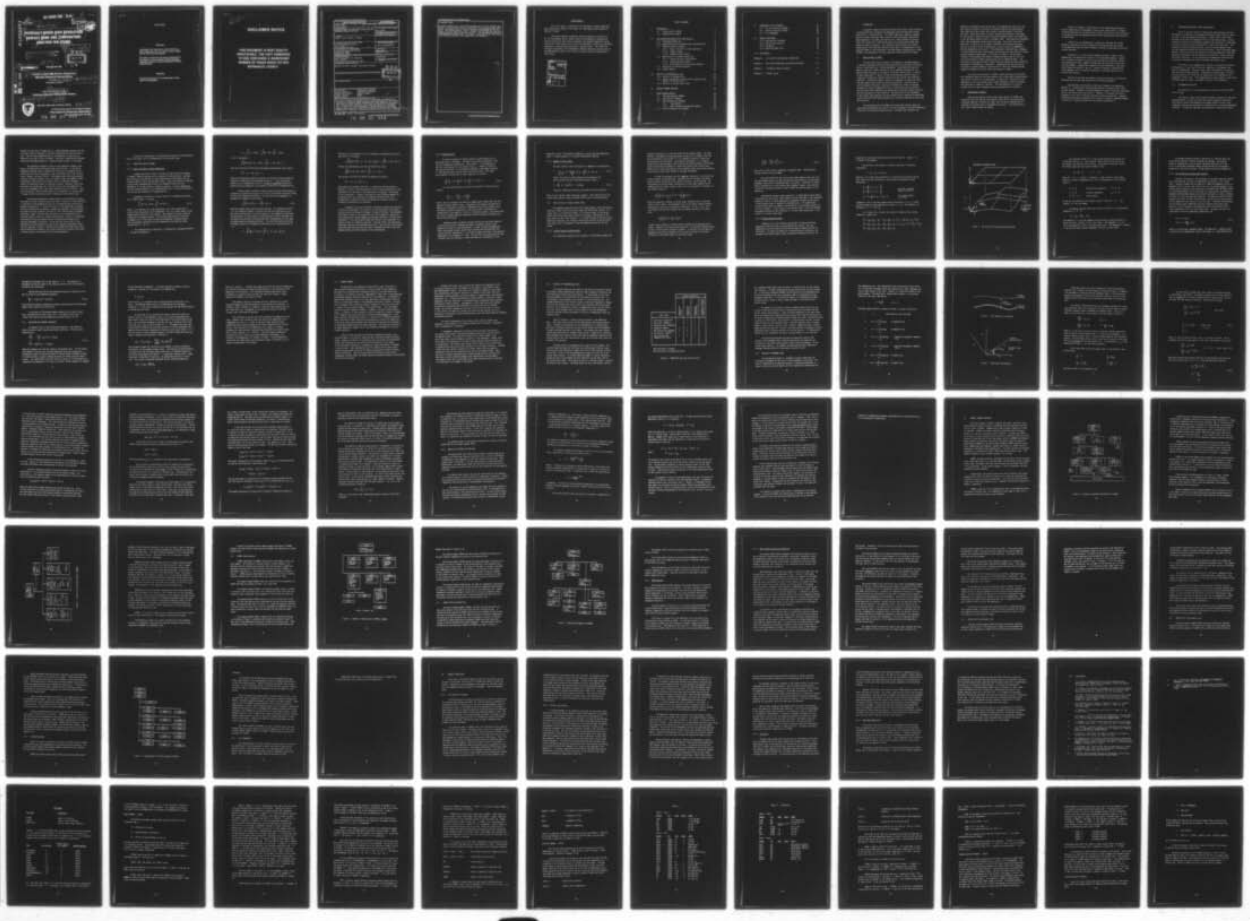
AD-A055 861 MISSION RESEARCH CORP SANTA BARBARA CALIF F/G 4/2
PHYSICALLY-BASED HIGH RESOLUTION SURFACE WIND AND TEMPERATURE A--ETC(U).
MAR 78 J A BALL, S A JOHNSON DAEA18-77-C-0043
UNCLASSIFIED MRC-R-7731-1-278 ERADCOM/ASL-CR-78-0043-1 NL

1 OF 3
ADA
055861



(19) ASL-CR-78-0043-1

(18) ERADCOM/ASL

AD
Reports Control Symbol
OSD-1366

(6)

PHYSICALLY-BASED HIGH RESOLUTION SURFACE WIND AND TEMPERATURE ANALYSIS FOR EPAMS.

(9) Final rept.,

(12) 278 p.

(11) MAR[redacted]78

Prepared By

(10) Joseph A. Ball and Steven A. Johnson
Mission Research Corporation
P.O. Drawer 719
Santa Barbara, CA 93102

(14) MRC-R-7731-1-278

(15) Under Contract DAEA 18-77-C-0043

Contract Monitor: Ronald M. Cionco

(16) 1L161102B53A (17) SA1

406 548

US Army Electronics Research and Development Command
Atmospheric Sciences Laboratory
White Sands Missile Range, N.M. 88002

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

The citation of trade names and names of manufacturers in this report is not to be construed as official Government indorsement or approval of commercial products or services referenced herein.

Disposition

Destroy this report when it is no longer needed. Do not return it to the originator.

DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DDC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ASL-CR-78-0043-1	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) PHYSICALLY-BASED HIGH RESOLUTION SURFACE WIND AND TEMPERATURE ANALYSIS FOR EPAMS		5. TYPE OF REPORT & PERIOD COVERED Final Report 6. PERFORMING ORG. REPORT NUMBER MRC Report #7732-1-278
7. AUTHOR(s) Joseph A. Ball & Steven A. Johnson		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Mission Research Corporation P. O. Drawer 719 Santa Barbara, CA 93102		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS DA Task 1L161102B53A/SA1
11. CONTROLLING OFFICE NAME AND ADDRESS US Army Electronics Research and Development Command Adelphi, MD 20783		12. REPORT DATE March 1978 13. NUMBER OF PAGES 274
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Commander/Director US Army Atmospheric Sci Laboratory ATTN: DELAS-D White Sands Missile Range, NM 88002		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

ABSTRACT

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Surface wind	Sub-mesoscale resolution
Surface temperature	Variational adjustment
Surface layer profiles	Computer program
Thermal structure	EPAMS
Complex terrain	

APPROX.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report documents the theoretical basis, development, and computational structure of a numerical computer analysis routine incorporated in the US Army Experimental Prototype Automatic Meteorological System (EPAMS) for the estimation of surface layer wind fields at sub-mesoscale resolution (≈100 meters) over a limited area in broken topography. The geographically re-locatable analysis exploits detailed topographic information but requires only limited meteorological information. The physically-based analysis uses Gauss' Principle

CONT.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20 ABSTRACT (cont)

of Least Constraints for a variational adjustment of an initial estimated wind field in a single surface layer to conform with terrain structure, mass conservation, and buoyancy forces. Fields of surface air temperature are also produced. Initial meteorological input is obtained from the EPAMS data base by an automated analysis which is described in detail. The segmentation structure of the computational program levels is presented. Appendices provide user instructions, detailed algorithms, and example wind field estimates.

ACKNOWLEDGEMENT

This final report is submitted to the Atmospheric Sciences Laboratory, White Sands Missile Range in fulfillment of a requirement under Contract No. DAEA 18-77-C-0043.

The Mission Research Corporation wishes to express its gratitude to Mr. William Ohmstede and Mr. Ronald Cionco of the Atmospheric Sciences Laboratory not only for their active cooperation and valuable suggestions during the course of this project, but also for the confidence to pursue the innovative approach to a surface layer wind analysis reported herein. The cooperation exhibited by personnel of the Atmospheric Sciences Laboratory and the staff of the WSMR Computing Center at all levels has greatly aided the performance of this project.

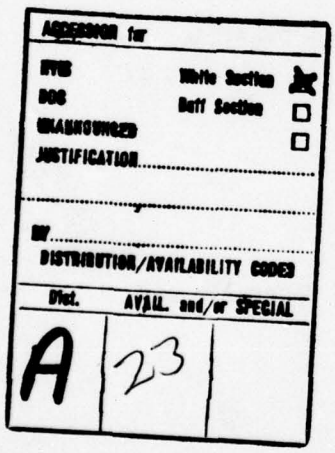


TABLE OF CONTENTS

I. INTRODUCTION

A successful method to determine surface wind fields at high resolution in varied conditions of weather and topography would find many applications supporting U.S. Army operations. This report describes a numerical computer analysis routine, developed under contract by Mission Research Corporation, whose objective is to supply high resolution ($\sim$ 100m) estimates of surface layer wind fields over a limited area taking into account both rugged topography and thermal structure. This numerical analysis which uses limited meteorological input data but detailed terrain data is incorporated as a subordinate element in the Experimental Prototype Automatic Meteorological System (EPAMS) at the Atmospheric Sciences Laboratory, White Sands Missile Range.

I.1 Scope of Work in EPAMS

The high resolution wind analysis implemented in the EPAMS embodies an innovative approach to meet the challenge posed by the problem of wind estimation at high resolution in broken topography. The method rests upon the physical equations of mass conservation and of momentum conservation, but employs these principles in the special variational form of Gauss' Principle of Least Constraints.[10] Gauss' Principle distinguishes external forces on a mechanical system from internal constraint forces arising from constraint conditions. According to Gauss' Principle the equations of motion are satisfied when these latter constraint forces are minimized. Starting from an initial estimate the analysis results are obtained by a direct variational relaxation of wind and temperature fields in the surface layer to minimize the constraint forces imposed by the warped terrain surface, thermal structure and the requirement of flow continuity. The application of this prodedure to the surface layer requires also the systematic use of empirically established surface layer profiles,[6] whose parameters are internally computed by the surface layer analysis.

For operation within the EPAMS the surface layer analysis needs not only detailed topographic data but also initial estimates of wind and temperature fields over the restricted simulation area. Although these estimates can

optionally be entered directly by the user, an automated data acquisition and data analysis procedure is provided to routinely generate the meteorological input required by the surface layer analysis from data stored in the EPAMS data base. These data consist of quality controlled observations (both upper air and surface), output from lower resolution mesoscale models, and predictions of the USAF Global Weather Central "fine-mesh" model. They embrace two geographic areas, the southwestern United States and Europe. The spatial resolution of these data is typically of the order of 100-200 km which requires the data analysis to bridge a factor of perhaps 50 in scale to obtain an estimate of wind and temperature at the small ($\sim$ 10km dimension) simulation area. Although a reliable mesoscale model could perhaps best perform this data analysis function, suitable mesoscale model output is not always available in the data base. Therefore, the automated data analysis produces estimates used for the local surface wind analysis without requiring the use of a mesoscale model. The diversity of data types and their varying relevance requires several alternative techniques or combinations of techniques. Consequently the data acquisition and analysis routines comprise a major portion of the total software package needed to implement the surface layer wind analysis in EPAMS.

The surface layer analysis in EPAMS is based upon physical principles and incorporates many well-verified features, but it does contain approximations the validity of which is not established. In particular, restriction of the computation to only the surface layer, without additional vertical structure, may constitute an important limitation. This new approach to a difficult problem area will require adequate testing and observational validation.

I.2 Organization of Report

There are two aspects of the surface layer analysis in EPAMS; the scientific basis for both the wind analysis itself and for the data analysis procedures required to furnish its input, and, the actual computational algorithms and structure built to implement the analysis. The organization of this report reflects these two aspects.

Section II presents a justification for the computational approach in terms of objectives and requirements of the surface layer analysis. Through a detailed discussion of the application of Gauss' Principle of Least Constraints to the surface layer the theoretical basis of the high resolution wind analysis is established.

Section III, also substantially theoretical, describes the various methods of data analysis used to determine from meteorological data stored in the EPAMS data base the input information required for operation of the local surface wind model.

Sections IV, V, VI describe the program structure in terms of logic, segmentation, and operational function of its constituent elements. Section IV is a general overview of the large scale structure while Section V details the organization of major segments and describes the functions of their subordinate elements. Computational segments are cross-referenced to the appropriate theory of Sections II and III. Section VI contains a description of some independent utility programs constructed to facilitate the use of the surface layer analysis.

Section VII provides some general conclusions relating to the analysis technique and describes some possibilities of future development.

Some important, but detailed, matters are treated in the appendices. Appendix A is a users manual containing descriptions of program files, inputs, and user options. Appendix B provides mathematical detail of the basic algorithms used by the surface layer wind analysis. Appendix C illustrates the results obtained by trial runs of the high resolution surface wind analysis. Appendix D completes the program documentation by providing a complete, commented FORTRAN listing of all programs developed for the surface layer analysis.

II. HIGH-RESOLUTION SURFACE LAYER WIND ANALYSIS

The basic analysis routine to produce the high-resolution wind estimate employs an extension of the concepts described in BALL (1975); an internal report of Mission Research Corporation. Starting from initial rough estimates of the wind and temperature fields in the surface layer, a direct variational relaxation is performed to minimize the "constraint" in the sense of Gauss' Principle of Least Constraints[10] as applied to fluid flows. The computation uses one constant-thickness layer, the volume between a user selected computation height and the terrain surface. Within the warped computational layer which follows the terrain the wind and temperature fields are adjusted as permitted by terrain geometry and conservation laws to minimize the dynamic constraint. The resulting wind and temperature fields provide an estimate conformable with input information and physical laws.

The choice of this somewhat novel approach is governed by objective considerations of input data limitations, operational requirements, and physical phenomenology. Below we discuss these considerations to show how they justify the method selected. Subsequently we discuss Gauss' Principle of Least Contraint for fluids and its application to the dynamics of the atmospheric surface layer. The implementation of these concepts in the finite difference numerical code is described in Section V.4 and Appendix B.

II.1 PERFORMANCE OBJECTIVES

The objectives of the sub-mesoscale wind analysis within the EPAMS are as follows.

The numerical model should provide horizontal resolution of the order of 60-400 meters, a scale far smaller than typical mesoscale resolution, sufficient to resolve the mechanical and thermal effects of small scale topographic features and variations in characteristics of the earth/atmosphere interface. In particular the model should operate in rough terrain where

slopes and changes of slope may be large and abrupt. Commensurate with the horizontal resolution the modeled area comprises a region only 5-20 km on a side or 25-400 square kilometers in area.

Only winds within the atmospheric surface layer adjacent to the terrain surface are desired. The vertical resolution of the model may be limited to the surface layer, no more than 50 meters above the ground.

The model should be applicable at arbitrary locations at any time of day and under a large class of prevailing meteorological conditions. It should be re-locatable geographically and as versatile as possible.

Temporal variation of the windfield within the limited spatial area of the simulation may be considered quasi-steady. The model itself need include no time-dependence but should produce windfields dependent upon input variables which themselves exhibit significant variation only over time intervals of one hour or more.

The model should be capable of operation with minimal meteorological input of much lower spatial resolution than the resolution of the model. Typically input data might consist of general stratification and estimates of general surface wind in the modeled area. These estimates may derive from any source; automated data analysis, mesoscale models of coarser resolution, or standard forecasts. In this specific application the estimates are derived by automated analysis of data resident in the EPAMS data base.

II.2 REQUIRED PHYSICAL AND COMPUTATIONAL CHARACTERISTICS

Achievement of the feature of general applicability requires that the high-resolution wind analysis rest as directly as possible upon the basic physics underlying the phenomena. The need for special computational parameterization, requiring extensive measurements or testing, decreases in proportion to the generality of the principles successfully embodied in the

calculation. Well established empirical parameterization of general applicability can, however, provide computational economies and can permit incorporation of detail difficult or impossible to model physically.

The feature of limited area allows an analysis based upon a steady-state, diagnostic application of the governing equations. A steady-state model can be structured to incorporate the significant terrain and thermal effects and to average transient, fine scale, time-dependent eddy structure. A spatially extrapolative/interpolative model is capable of incorporating the effects of local inhomogeneities upon the wind field. Temporal dependence over time intervals exceeding one hour can be provided by alteration of meteorological input parameters.

The model structure should reflect the lack of resolution and detail in meteorological input. A rational model must be structured to utilize maximally the available input without either explicitly or implicitly assuming more information than the input provides. It should have the character of an estimation in a limited information situation. Detailed topographic information, the physical laws of fluid dynamics, general empirical surface layer profiles, plus limited meteorological input data constitute the information to be used. The conventional approach of solving a boundary value problem requires assumptions of boundary values which are not adequately provided by the input information. In a situation of limited information direct variational methods, which incorporate the physical phenomenology but which do not require solution of a boundary value problem, seem particularly appropriate. Both boundary and interior values of wind and temperature can be varied to optimize the fields consistent with the constraints imposed by input data, physical laws, atmospheric stratification and topographic structure. A variational model will not produce an exact or unique solution, but rather a useful estimate conformable with the available information.

Terrain sheltering and channeling, wakes, and flow separation are

features of wind flow in rugged terrain. These phenomena originate from the non-linear inertial character of the flow interacting with the terrain surface. The non-linear advective terms of the equations of motion can be dominant at the small scales considered. Therefore a computational procedure based upon the momentum equation, including advective terms, is indicated.

The turbulent character of flows in the atmospheric boundary layer poses a challenge which any physical model must confront. If the entire atmospheric boundary layer is modeled the inclusion of turbulence and its effect upon the flow is unavoidable. Near the surface, within the "surface layer", however, vertical fluxes of fluid momentum carried by the turbulent stresses are nearly constant. See, for example, LUMLEY and PANOFSKY (1964). A computation restricted to the surface layer can therefore reasonably neglect the momentum transfer of turbulent stresses. Moreover, there exist experimentally derived empirical profiles of mean wind and temperature in this layer, BUSINGER (1973), for steady conditions over flat terrain. These profiles applied, either by modification or by assumption, to curved flows in rugged terrain can provide vertical resolution of the surface layer.

The selected computational method embodies the considerations above. The method of direct variation based upon Gauss' Principle of least constraint is equivalent to the momentum equation and incorporates both advective and buoyancy accelerations. A direct variation of field quantities, consistent with mass conservation and potential temperature advection, at both boundary and interior points eliminates the need for prior specification of boundary conditions. Gauss' Principle does not involve a time integration and is therefore ideally suited to a steady-state estimate. Choice of a coordinate system which follows the warped terrain surface permits maximum use of detailed topographic information. Use of only the surface layer in the computation, the most doubtful and restrictive feature; nevertheless simplifies the equations, permits an essentially two-dimensional model with consequent computational economies, and focuses attention on surface effects which at these scales are the dominant wind field perturbations.

We return to these points in the detailed discussion of the theoretical basis of the model, and its implementation to the surface layer.

II.3 THEORETICAL BASIS OF MODEL

II.3.1 Gauss' Principle of Least Constraints

A modern discussion of Gauss' Principle of Least Constraints appears in LANCZOS (1962) but applies only to systems of point particles. Long ago, the relation of the principle of least constraints to fluid motion was considered by APPELL (1912) and GUILLAUME (1913). Since the principle as applied to non-viscous incompressible fluids, the assumptions we use for the atmosphere, is not well known; we review its details. First we demonstrate that the pressure gradient in an incompressible fluid is a force of constraint arising from the constraint of incompressibility.

D'Alembert's principle of virtual work for an incompressible perfect fluid, SERRIN (1959), is written,

$$\int_V \rho(\vec{A} + \vec{g}) \cdot \delta\vec{x}\, dV - \int_S t\ \delta\vec{x} \cdot d\vec{S} = 0 \qquad \text{(II.1)}$$

where ρ is the density, $\vec{A}$ is the acceleration, $-\vec{g}$ is the acceleration of gravity (the only external force considered), and t is a scalar stress in the outward normal direction to the surface. The integrals are over a material volume and its boundary surface. The principle requires that virtual displacements $\delta\vec{x}$ of a fluid particle obey the kinematical constraints which here are boundary constraints and the incompressibility condition (mass conservation), $\nabla \cdot \delta\vec{x} = 0$.

The incompressibility constraint is introduced by a Lagrange multiplier λ. We add the expression,

$$0 = -\int_V \lambda \nabla \cdot \delta\vec{x}\, dV = -\int_S \lambda \delta\vec{x} \cdot d\vec{S} + \int_V \nabla\lambda \cdot \delta\vec{x}\, dV$$

to (II.1) and obtain,

$$\int_V [\rho(\vec{A} + \vec{g}) + \nabla\lambda] \cdot \delta\vec{x}\, dV - \int_S (t + \lambda)\, \delta\vec{x} \cdot d\vec{S} = 0.$$

Since the variations $\delta\vec{x}$ may now be considered unconstrained, there results:

$$\rho\vec{A} = -\nabla\lambda - \rho\vec{g} \quad \text{and} \quad t = -\lambda.$$

These are the familiar equations of motion for inviscid fluids provided the pressure is identified as the Lagrange multiplier λ. Thus the pressure gradient in an incompressible fluid is a force of constraint necessary to enforce the pure kinematical (not dynamical) condition of incompressibility.

In words, Gauss' Principle of least constraint asserts that motion of a mechanical system takes place in such a way as to minimize constraint forces arising from kinematical conditions. For our non-viscous atmosphere its mathematical statement can be written,

$$\delta \int_V \frac{\rho}{2} (\vec{A} + \vec{g})^2\, dV - \delta \int_S t\vec{A} \cdot d\vec{S} = 0 \qquad \text{(II.2)}$$

where the symbols retain their previous meanings. The indicated variation, δ, of the integrals denotes a variation in which only accelerations, not positions and velocities, are varied subject to the kinematical constraints. The equivalence of this principle to the equation of motion, provided the variations of acceleration are consistent with incompressibility, may again be demonstrated by Lagrange multipliers. The total time derivative of the divergence condition (a kinematical constraint) with a Lagrange multiplier λ is:

$$0 = -\int_V \lambda \frac{D}{Dt} (\nabla \cdot \vec{v})\, dV = -\int_V \lambda\, (\nabla \cdot \vec{A} - v^i_{,j}\ v^j_{,i})\, dV$$

Addition of this expression to (II.2) followed by integration by parts of the term in $\nabla \cdot \vec{A}$ yields,

$$\delta \int_V \left[\frac{\rho}{2} (\vec{A} + \vec{g})^2 + \nabla\lambda \cdot \vec{A} + \lambda v^i_{,j} v^j_{,i}\right] dV - \delta \int_S (t + \lambda)\, \vec{A} \cdot d\vec{S} = 0.$$

Varying the accelerations, but not the velocities, gives,

$$\int_V \left[\rho(\vec{A} + \vec{g}) + \nabla\lambda\right] \cdot \delta\vec{A}\, dV - \int_S (t + \lambda) \cdot \delta\vec{A} = 0.$$

The arbitrary variation $\delta\vec{A}$ implies the equations of motion,

$$\rho\vec{A} = -\nabla\lambda - \rho\vec{g} \quad \text{and} \quad t = -\lambda$$

where again the Lagrange multiplier λ is identified with the pressure. Equation (II.2) has a positive quadratic term in the accelerations, so the vanishing of its variation is the condition for a minimum. The minimum of the "constraint" -- taken as the square of the difference between the acceleration and the acceleration of external applied forces ($-\vec{g}$) -- characterizes a situation in which the equations of motion, as well as kinematical constraints, are satisfied.

Customarily one uses variational principles to derive the equations of motion, as done above. By contrast, the high resolution wind analysis directly varies the constraint integral to seek its minimum. Both boundary and internal values of flow quantities are varied subject to constraints -- mass conservation, surface geometry and boundary conditions, limited input information. Since the principle does not involve a time integration it is ideally suited to a diagnostic analysis. Pressure forces, the constraints minimized, need never be considered explicitly. But the final wind estimate which minimizes the constraint provides an estimated solution of both the momentum equation and the equation of mass conservation.

II.3.2 Thermal Effects

In order to account for thermal effects an approximation of the Boussinesq type is introduced. The effective external acceleration is not $-\vec{g}$, but the buoyancy acceleration $-\rho'\vec{g}/\rho_o$ where ρ' is the departure of the local density away from its ambient value ρ_o. Likewise in place of the external pressure, -t, the effective pressure is the dynamic pressure, P', the departure away from the static equilibrium value. With these approximations the principle of least constraint is written,

$$\delta\int_V \frac{\rho_o}{2}\left[\vec{A}+\frac{\rho'\vec{g}}{\rho_o}\right]^2 dV + \delta\int_S P'\,\vec{A}\cdot d\vec{S} = 0. \qquad \text{(II.3)}$$

To a good approximation the relative fluctuation in density may be written,

$$\frac{\rho'}{\rho_o} \simeq -\frac{(T-T_o)}{T_o} \simeq \frac{-(\Theta-\Theta_o)}{\Theta_o}$$

where T and Θ are absolute and potential temperatures respectively, and zero subscripts denote ambient values. Consequently the buoyancy acceleration is modeled by fluctuations of potential temperature away from an ambient value. The ambient potential temperature in the surface layer is determined by extrapolation of the upper air profile to the ground surface. Once established the field of ambient potential temperature remains unchanged throughout the rest of the calculation.

Buoyancy forces are however neither constant nor accurately known as a function of position, although they appear as external forces in the dynamical principle of least constraints. In order to make the temperature field, and buoyancy forces, advectively consistent with the windfield the surface potential temperature, Θ, is varied to reduce the value of the potential temperature flux divergence, $\nabla\cdot(\vec{v}\Theta)$. This auxiliary variation does not seek a minimum, but only operates to reduce the advective inconsistency of the

temperature field. The potential temperature, rather than the temperature itself, is chosen because it is the more conservative quantity.

II.3.3 Summary of Basic Theory

The basic theory of the wind analysis is embodied in the equations;

1. $$\delta \int_V \frac{\rho_o}{2} \left[\vec{A} - \frac{(\theta - \theta_o)\vec{g}}{\theta_o} \right]^2 dV + \delta \int_S P' \vec{A} \cdot d\vec{S} = 0. \qquad \text{(II.4)}$$

(Gauss' Principle of Least Constraints with buoyancy and surface dynamic pressure considered external forces.)

2. $$\int_V [\nabla \cdot (\vec{v}\theta)]^2 \, dV \rightarrow \text{minimum.} \qquad \text{(II.5)}$$

(Potential temperature advection consistent with the windfield)

These are not the basic model equations, however. Their application to the surface layer analysis requires further modifications as discussed below.

II.4 APPLICATION TO A SINGLE SURFACE LAYER

The modeled volume consists of the volume between a user selected computational height (within the surface layer) and the terrain surface. This single layer is of constant thickness in a direction normal to the terrain surface and follows the warped surface of the ground. Application of the basic theory to this layer requires: (1) A simplification of the basic three-dimensional equations, (2) A special non-Euclidian terrain following coordinate system, and (3) Estimates of wind and temperature profiles near the ground.

II.4.1 Surface Parallel Simplification

The theoretical equation (II.4) contains in the surface integral the

pressure fluctuation P' on the top surface of the modeled volume. The lower surface of the modeled volume contributes nothing to this integral since the no-slip condition at the ground, $\vec{v} = 0$ (a boundary contraint), forces the acceleration to be zero there. The pressure fluctuation, P', an external force arising from atmospheric motion outside the modeled volume, unfortunately cannot be accurately estimated in a model restricted to the surface layer. There exists an open upper boundary of the modeled volume across which pressure forces couple surface layer motion to the upper atmosphere.

In these circumstances the only modeling alternative is to discard the surface integral on the open face. One effect of this integral, the hydrostatic pressure fluctuation due to density variation, can be retained. The contribution of surface normal accelerations to the volume integral may be combined with the surface integral itself for a thin sublayer of thickness $\Delta\xi$. The relevant terms are;

$$\Delta\xi \int_S \frac{\rho_o}{2} \left[A_n + \frac{\rho'}{\rho_o} g_n \right]^2 dS + \Delta\xi \int_S \frac{\partial P'}{\partial \xi} A_n \, dS$$

where the subscript n denotes a surface normal component and ξ is position normal to the surface. The hydrostatic part of the normal pressure gradient fluctuation is just the local density fluctuation: $\partial P'/\partial\xi = -\rho' g_n$. With this substitution the two integrals may be combined resulting in a single term,

$$\Delta\xi \int_S \frac{\rho_o}{2} \left[A_n^2 + \left(\frac{\rho'}{\rho_o} g_n\right)^2 \right] dS$$

in which cross products of surface normal acceleration and buoyancy do not appear. Absence of the cross product means that surface normal buoyancy plays no role in the variation and may be ignored. On the basis of this heuristic argument in the place of equation (II.4) we use as the basic variational principle for the dynamic model of the surface layer,

$$\delta \int_V \left[\vec{A} - \frac{\Theta - \Theta_o}{\Theta_o} \vec{g}_{||} \right]^2 dV = 0 \qquad \text{(II.6)}$$

where only surface parallel components of buoyancy enter. Multiplicative constant factors are also dropped.

The justification for use of this basic model equation rests not only upon the argument above, but also upon the more structured and, by qualitative judgement, more realistic windfields which it produces. Inclusion of surface *normal acceleration, but not surface normal buoyancy*, appears to be required in order for the model to exhibit terrain sheltering and some effect of flow separation.

Nevertheless the basic model equation is an approximation which effects a de-coupling of the surface layer analysis from the remainder of the atmosphere. Despite its use of all three acceleration components the model restricted to a single surface layer is essentially two-dimensional. The computational simplicity thus gained is bought at the expense of a well-founded analysis of vertical structure. In the course of this project several simplified attempts were made to improve on this situation, all of which were unsuccessful in the context of a surface layer model.

II.4.2. Warped Coordinate System

Equation (II.6) still implies a great deal of structure from the adaptation of the wind field to the warped terrain surface. Near the impenetrable terrain surface the wind velocity must be largely parallel to the surface. To account accurately for convoluted terrain we employ a non-orthogonal coordinate system in which coordinate directions and scales are

determined by the terrain structure and vary with location. Figure 1. is a diagram of the geometry.

Any position on the surface is given by the vector in Cartesian coordinates,

$$\vec{r} = \hat{i}x + \hat{j}\,y + \hat{k}\,h(x,y)$$

where h(x, y) is the surface elevation as a function of horizontal location. Base vectors parallel and normal to the surface are obtained by taking derivatives.

$$\left.\begin{aligned} \vec{a}_1 &= \frac{\partial \vec{r}}{\partial x} = \hat{i} + \hat{k}\,h_x \\ \vec{a}_2 &= \frac{\partial \vec{r}}{\partial y} = \hat{j} + \hat{k}\,h_y \end{aligned}\right\} \quad \text{parallel to surface (not unit vectors)}$$

$$\hat{a}_3 = \hat{n} = \frac{1}{\sqrt{a}}(-\hat{i}\,h_x - \hat{j}\,h_y + \hat{k}) \quad \text{unit vector normal to surface.}$$

Alphabetic x and y subscripts denote partial derivatives, i.e., $h_x = \partial h/\partial x$, and unit vectors are distinguished by a caret, i.e., $\hat{a}_3$. The quantity a is defined as: $a = 1 + h_x^2 + h_y^2$.

It is useful also to define the system of reciprocal base vectors denoted by a superscript.

$$\vec{a}^1 = (\vec{a}_2 \times \vec{a}_3) \,/\, [\vec{a}_1 \cdot (\vec{a}_2 \times \vec{a}_3)] = \frac{1}{a}\,[\hat{i}\,(1 + h_y^2) - \hat{j}\,h_x h_y + \hat{k}\,h_x]$$

$$\vec{a}^2 = (\vec{a}_3 \times \vec{a}_1) \,/\, [\vec{a}_1 \cdot (\vec{a}_2 \times \vec{a}_3)] = \frac{1}{a}\,(-\hat{i}\,h_x h_y + \hat{j}\,(1 + h_x^2) + \hat{k}\,h_y]$$

$$\hat{a}^3 = (\vec{a}_1 \times \vec{a}_2) \,/\, [\vec{a}_1 \cdot (\vec{a}_2 \times \vec{a}_3)] = \hat{a}_3$$

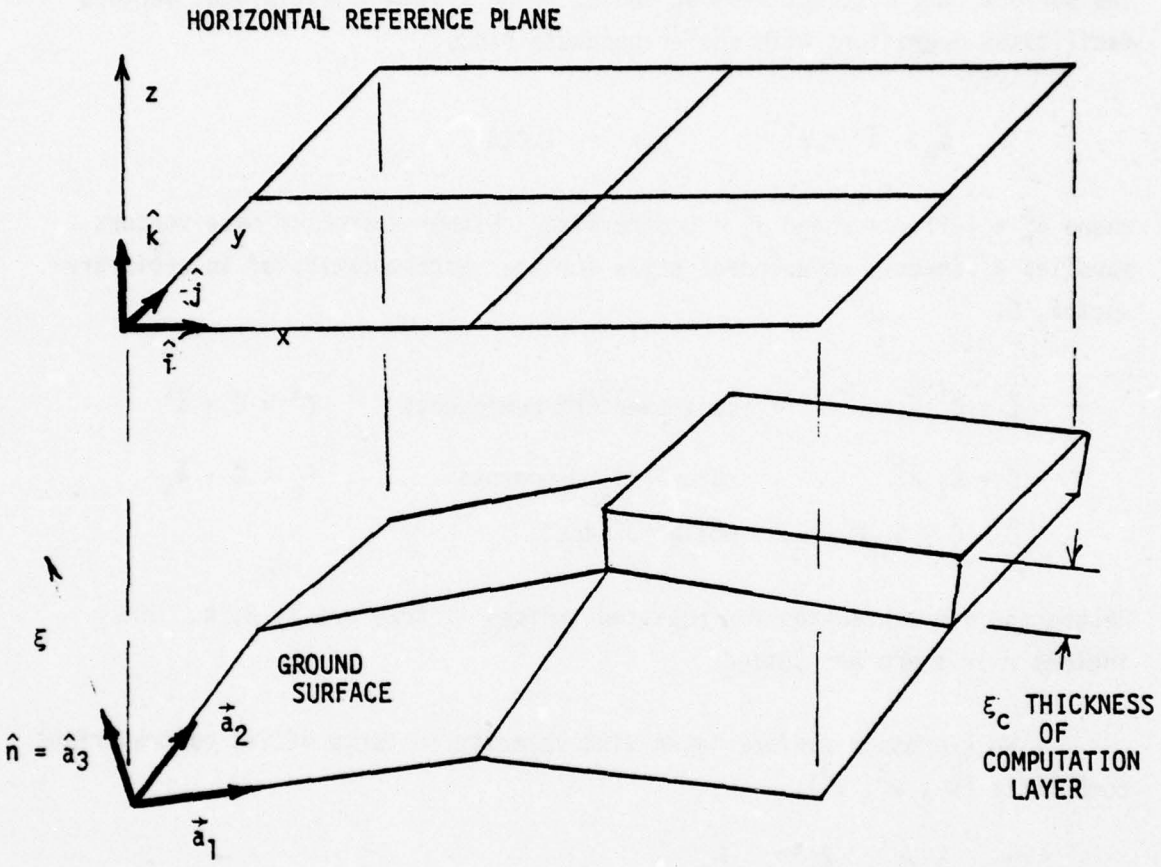


Figure 1. The Terrain Following Coordinate System

The reciprocal vector $\hat{a}^3$ is a unit vector normal to the surface and is identical with $\hat{a}_3 = \hat{n}$. The reciprocal vectors $\vec{a}^1$ and $\vec{a}^2$ are parallel to the surface, but distinct from $\vec{a}_1$ and $\vec{a}_2$. The system of reciprocal vectors facilitates operations with scalar products since,

$$\vec{a}_r \cdot \vec{a}^s = \delta_r^s \qquad r,s = 1,2,3$$

where $\delta_r^s = 1$ if $s = r$ and $\delta_r^s = 0$ otherwise. Either system of base vectors supplies a linearly independent basis for the representation of an arbitrary vector, $\vec{C}$.

$\vec{C} = C^i \vec{a}_i$	contravariant components	$C^s = \vec{C} \cdot \vec{a}^s$
$\vec{C} = C_i \vec{a}^i$	covariant components	$C_s = \vec{C} \cdot \vec{a}_s$
$\vec{C} \cdot \vec{C} = C_i C^i$	scalar product	

We use the sum convention for repeated indices if they are i, j, k. The indices r or s are not summed.

We express a surface layer wind velocity in terms of its contravariant components (v^1, v^2, v^3).

$$\vec{v} = v^1 \vec{a}_1 + v^2 \vec{a}_2 + v^3 \hat{a}_3$$

The component v^1 is not the component of wind velocity along the surface in the $\vec{a}_1$ direction, since $\vec{a}_1$ is not a unit vector. The component v^1 is, in fact, the projection of $v^1 \vec{a}_1$ in the horizontal plane. The component v^3, however, is the surface normal velocity component, since $\hat{a}_3$ is a unit vector.

The warped coordinate system is defined at all locations where the horizontal derivatives of terrain elevation are defined. The formulas above are systematically used to express the model equations (II.5) and (II.6) in the warped coordinate system, thereby building in the geometric properties of the terrain surface. Further details are given in Appendix B.

II.4.3 Use of Empirical Surface Layer Profiles

Accurate calculation of the integrands of the model equations requires use of surface normal profiles of temperature and velocity in the surface layer. The model assumes that the empirical, stability dependent profiles of BUSINGER (1973) valid over level terrain may still be applied locally in the surface normal direction at the resolution of the model even for rough terrain. To account approximately for flow curvature in rough terrain, the analogy of BRADSHAW (1969) between buoyancy and streamline curvature is used to estimate the effects of flow curvature upon stability. The empirical profiles require values at two levels for their specification, but the surface layer analysis carries temperature and velocity at only one level. Therefore an approximate bulk Richardson number determined from flow curvature and surface layer buoyancy is used to determine an approximate power law wind profile and a linear temperature profile in the surface layer. The model profiles to be determined for wind and temperature respectively are,

$$u(z) = b\,(z/z_o)^n$$
$$\Theta(z) = \Theta_c + \left.\frac{d\Theta}{dz}\right)_c (z-z_c) \qquad \text{(II.7)}$$

where z_o is the surface roughness length. The subscript, c, denotes values at the top of the computational layer of thickness, z_c. The coefficient and

exponent of the wind profile are determined by matching the logarithmic derivative and value of the wind speed at z_c to the empirical wind profile. The vertical derivative of temperature is obtained by forcing an approximate equality between the gradient and bulk Richardson numbers in the surface layer.

We assume that gradient and bulk Richardson numbers are approximately equal,

$$Ri_t = g(d\Theta/dz) \; / \; \Theta_o\left(\frac{du}{dz}\right)^2 \simeq g\frac{(\Theta_o - \Theta)}{\Theta_o}\frac{z}{u^2} \qquad \text{(II.8)}$$

where all quantities are evaluated at the top of the computational layer. Variables carried in the analysis permit computation of the bulk Richardson number. This Richardson number is modified by adding to it an analog of the gradient Richardson number for curved flows. BRADSHAW (1969) gives this analog as,

$$Ri_{ic} = 2\,S(1 + S) \quad \text{where} \quad S = (u/r) \; / \; (du/dr)$$

and r is the radius of streamline curvature (positive for curvature convex upward). Using the assumed power law profile and the average surface normal acceleration A_n in the layer, and assuming A_n is purely centripetal, S can be approximated.

$$S = -\frac{2z_c A_n}{u_c^2}$$

Addition of this term to the thermal Richardson number provides a stability measure accounting for flow curvature.

$$Ri = Ri_t + Ri_{ic}$$

The surface normal acceleration enters the model not only directly in the basic variational equation (II.6) but also indirectly through its effect upon the stability dependent profiles.

The empirical wind shear and wind profiles of BUSINGER (1973) are:

$$\frac{kz}{u_*}\frac{du}{dz} = \phi_m = (1-15\xi)^{-4}; \quad \frac{u}{u_*} = \frac{1}{k}\left(\ln\frac{z}{z_o} - \psi_1\right); \quad \xi < 0, \; Ri < 0$$

$$\frac{kz}{u_*}\frac{du}{dz} = \phi_m = (1 + 4.7\xi); \quad \frac{u}{u_*} = \frac{1}{k}\left(\ln\frac{z}{z_o} + 4.7\xi\right); \quad \xi > 0, \; Ri > 0$$

where $\xi = z/L$, the height scaled by the Obukhov length. The function ψ_1 is,

$$\psi_1 = 2 \ln\left[(1+x)/2\right] + \ln\left[(1+x^2)/2\right] - 2\tan^{-1}(x+\pi/2)$$

where $x = (1 - 15\xi)^{-1/4}$.

We evaluate the function ϕ_m at the layer top according to the value of the Richardson number in the range, $-5.0 \leq Ri \leq 0.25$. If Ri is less than the lower limit, it is set equal to the lower limit, -5.0.

$$\begin{aligned}
\phi_m &= (1-12Ri)^{-1/4}; && Ri \leq 0.0 \\
&= (1-3Ri)^{-1}; && 0.0 < Ri < 0.03571 \\
&= 0.88\,(1-6Ri)^{-1}; && 0.03571 \leq Ri \leq 0.1246 \\
&= 1.75\,(1-4Ri)^{-1}; && 0.1246 < Ri \leq 0.25
\end{aligned}$$

Matching the logarithmic derivative between the empirical profiles and the assumed power law profiles provides the value of the exponent in the power law profile.

$$n = \phi_m \Big/ \begin{cases} \left(\ln\frac{z}{z_o} - \psi_1\right) & \xi < 0 \\ \left(\ln\frac{z}{z_o} + 4.7\xi\right) & \xi > 0 \end{cases} \qquad \text{(II.9)}$$

The power law exponent falls in the range $0 < n < 1$. The constant b is determined by the wind speed at the layer top, which is the field quantity carried by the calculation.

Knowing the wind profile the approximate equality of equation (II.8) may be solved for the temperature gradient.

$$\frac{d\theta}{dz} = 2(\theta_o - \theta)\, n^2 / [z(2-n^2)] \qquad \text{(II.10)}$$

This procedure produces a temperature gradient consistent with the Richardson number used to obtain the wind profiles.

In the case of a Richardson number exceeding 0.25, stable laminar flow, the empirical profiles are not valid. We assume in this case $n = 1$ (linear wind profile) and the temperature gradient given by (II.10).

II.5 RELAXATION FOR MINIMUM CONSTRAINT

An essential part of the surface wind analysis is the method of direct relaxation used to obtain the minimum constraint. The basic model equations are;

$$\int_V [\vec{A} - \frac{\theta-\theta_o}{\theta_o}\, \vec{g}_{||}]^2\, dV = \text{minimum}$$

$$\int_V [\nabla \cdot (\vec{v}\theta)]^2\, dV \rightarrow \text{minimum} \qquad \text{(II.11)}$$

where the integrals are over the volume of the modeled layer. For the steady state model $\vec{A}$ includes only the time-independent advective acceleration. The modeled volume is divided into boxes of surface normal thickness z_c whose horizontal profile is square although the boxes are warped with the terrain surface. The contribution of each of these flux boxes, so called because the integrands are expressed in flux form, to the total volume integrals is computed

by the algorithms of Appendix B. The total constraint integral can be expressed as a sum over all flux boxes in the modeled area.

$$R_T = \sum_{i,j} R_{ij}$$

Each of the R_{ij} is a function only of surface parallel velocities on its faces. (The surface normal velocity is determined by the divergence condition, $\nabla \cdot \vec{v}$, the no-slip condition at the ground, and the normal profiles of surface parallel wind).

We associate an R_{ij} with each grid point at which the velocity is defined. The R_{ij} is the average of the R_{ij}'s of the two overlapping flux boxes upwind of the grid point, which both contain the velocity point on one face. For variational purposes we consider this R_{ij} a function only of the velocities (v^1_{ij}, v^2_{ij}) at the grid point (i, j). At boundary and corner grid points upwind flux boxes are not always possible, so we use contiguous ones. Denoting the two velocity components at each grid point by the index k = 1, 2; the unit vector of global steepest descent in the multi-dimensional velocity space to decrease the constraint is written as:

$$n^k_{ij} = -(\partial R_{ij}/\partial v^k_{ij}) \Big/ \left[\sum_{i,k,j} (\partial R_{ij}/\partial v^k_{ij})^2\right]^{1/2}$$

Each relaxation sweep over the entire grid computes a velocity correction proportional to the steepest descent vector. The constant of proportionality is chosen to fix the root mean square correction at a selected fraction of the initial approximate wind velocity -- a convenient velocity scale. Underrelaxation is required for stability. An appropriate fraction is 0.03 or 0.05. The velocity corrections generated at each relaxation sweep are;

$$\Delta v^k_{ij} = 0.03 v_o \sqrt{NxM}\, n^k_{ij}$$

where v_0 is the initial constant wind speed and N and M are the grid dimensions. All velocity corrections are applied simultaneously at the conclusion of the sweep over all grid points. This relaxation procedure apportions larger corrections to regions of the grid where the constraint integral is most sensitive to changes in the velocity field.

The numerical analysis calculates the total constraint R_T at each relaxation sweep and saves the windfield of the minimum value achieved. A minimum is usually reached in a number of relaxation steps roughly equal to the linear grid size. Subsequent to the minimum the constraint exhibits irregular small amplitude fluctuations.

The relaxation of the velocity field considers the temperature field fixed. However, each relaxation step also adjusts the temperature field independently by use of the second of the basic model equations above. Consequently each relaxation sweep yields altered temperature and velocity fields. The relaxation of the temperature flux integral is directly analogous with the constraint relaxation described above but simpler since only a scalar quantity, Θ, is involved. A root mean square temperature correction of 0.1K is used to scale the temperature adjustments. No minimum of the temperature flux integral is sought. Instead temperature corrections are simply made at each relaxation sweep to continually try to reduce the temperature flux integral.

II.6 GENERAL REMARKS

As discussed in the beginning of the section, Gauss' Principle of Least Constraints potentially offers a solution to many of the problems associated with a high-resolution diagnostic analysis of surface layer winds. It provides a variational principle, independent of time, which incorporates both the momentum and mass continuity equations of fluid mechanics. An essentially steady-state diagnostic analysis is often performed in meteorological applications by a direct application of time dependent equations of motion. Starting from an assumed initial state the system is advanced in time until some state is reached in which further changes in time become small. This state whose selection requires subjective judgment is deemed a "balanced" steady dynamic state. In contrast to this procedure in which the equation of motion is obeyed at all stages of the calculation the variational method searches for a state which satisfies the equation of motion. In the case of a variation based upon Gauss' Principle of Least Constraints the equations of motion are obeyed only in the final state which minimizes the constraint and then only to the degree of approximation permitted by the model algorithms. The selection criterion is no longer subjective, however, but is the objective attainment of a state of minimum constraint.

Since far from the constraint minimum the momentum equation is not satisfied there is no intrinsic reason why mass conservation need be enforced either. However, variations of acceleration near the constraint minimum must satisfy mass conservation, and the use of this principle at all stages in the variational relaxation is convenient to reduce the degrees of freedom in the variation. Mass conservation plus the velocity zero at the surface enables surface normal velocities to always be expressed in terms of surface parallel velocities. Thus the surface parallel velocities alone are the independent variables whose variations operating through the advective acceleration minimize the constraint.

Although the surface layer wind analysis employs the temperature field to estimate buoyancy forces, the emphasis of the model is upon dynamic rather than thermodynamic effects. Even within the limitations imposed by the surface layer approximation the dynamic constraint is handled in more detail than the thermodynamic energy equation, which only appears in reduced form as the potential temperature flux. Situations dominated by inertial effects - high winds, rough topography, and fine resolution - are probably better represented than situations dominated by diabatic heating. Improvements in this latter area could perhaps be achieved by accounting more accurately for sources and sinks of thermal energy, carrying both temperatures and wind velocities at two levels in order to fully exploit the empirical surface layer profiles (see Section II.4.3), or developing a variational principle based primarily upon the energy equation instead of the momentum equation.

An implicit assumption of the surface layer analysis, nowhere else mentioned, is the neglect of all effects of water and air moisture. No evaporation or condensation is considered.

A fundamental theoretical difficulty with an analysis restricted to the surface layer alone is the pressure boundary condition on the top face of the modeled volume. This problem is present in all dynamic models, even multi-layer ones, which include only a portion of the atmosphere. Models which only enforce flow continuity still need an upper boundary condition though not on the pressure. The surface layer analysis simply neglects the relevant term, a procedure which may be justified a posteriori by the results achieved. Although a model with more vertical structure (or layers) would likely be superior, assumptions would still be required at the top of the modeled volume. There may exist an optimum compromise which includes enough vertical structure but which does not make inordinate demands upon computational time and memory. The present single layer surface model with rather detailed vertical resolution is a reasonable first attempt at the optimum.

III. ANALYSIS OF METEOROLOGICAL DATA

This section describes the theory of the analysis required to extract from EPAMS resident meteorological data the input parameters required by the high-resolution surface layer wind analysis. In the absence of real-time spot measurements in the simulation area the quantities required are: (1) A background, uniform general wind estimate for the simulation area. This somewhat hypothetical wind takes no account of detailed topography or thermal structure but provides the initial field to be relaxed by the surface layer analysis; (2) An estimate of surface air temperature fields over the simulation site, which together with; (3) An estimate of vertical temperature profiles; provides starting estimates of surface air temperature fields and buoyancy forces.

Figure 2 displays the types of data available in the EPAMS Data Base. These data always contain surface observations and sounding observations. In addition the data base of the Southwestern U.S. contains USAF GWC hourly fine-mesh predictions and may contain output from the Atmospheric Sciences Laboratory mesoscale layer model. The European data base contains neither GWC predictions nor mesoscale model output. Figure 2 also indicates other potential data sources; direct user input of parameters required by the surface analysis and local observations within the simulation area, which are presently not available.

The data analysis requirements pose a very difficult problem. The available data are typically from sites up to 200 kms away from the simulation site with high intervening ridges, and up to 12 hours previous to simulation time. Errors of driving parameters can remove all validity of the surface layer analysis. A reliable meso-scale model is clearly required to bridge the gap in scale between the data resolution and the small area of the surface layer model. Construction and use of a mesoscale model is outside the scope of this project. The EPAMS system, we note, does however provide

Data Types	Output Information			
	Temperature Profile	Surface Air Temperature	Background Surface Wind	Structure of Local Temperature & Wind*
User Input (Specified)	1	1	1	
Local Obs. (MESNET)*		2	2	2
Mesoscale Model (MESMOD)+			3	
Soundings (MRCUA)	2	3	4	
GWC Predictions (MRCGWC)+	2	3	4	
Surface Obs. (MRCSFC)		3	4	
Data Insufficient (Request User Input)	3	4	5	

*Not Available at Present

+Not Available in European Data Base

Figure 2. EPAMS Data and Data Use Priorities

for a mesoscale model which could be used as a routine driver for the surface layer analysis. In these circumstances we cope with the data analysis problem by: 1) providing a mechanism whereby surface analysis driving parameters may be directly entered by the user at execution time, 2) preferentially using mesomodel output when it is available and timely, or 3) generating the best estimate possible, short of using a mesoscale model, on the basis of the available data.

The numbers in the chart of Figure 2 represent the priorities of use of each of the specified data types to provide the specified output required for the surface layer analysis. Equal priorities appearing in the same column denote the several data types which may be conjunctively required at that priority level to generate the information which labels the column. Insufficient data, or failure to find suitable data, results in a data analysis failure, the only remedy for which is user supplied input. With user supplied input, an option permits a wind analysis in which no use is made of temperature data, although dummy data must be supplied. Failure of the data analysis to generate a surface background wind will prevent execution of the wind analysis. A climatological default for this fundamental and highly variable quantity is not justified.

The following paragraphs describe the theoretical basis of the analysis of ASL mesomodel data, sounding observations and numerical prediction data, and surface station data to obtain driving parameters of the surface layer analysis. The somewhat involved software logic required to implement the data analysis is described in Section IV and V.

III.1 ANALYSIS OF MESOMODEL DATA

The ASL Mesoscale Model is a diagnostic analysis based upon the variational adjustment of the flow in a layer adjacent to the ground. The layer is identified by an objective analysis of sounding observations at several localities, and the results are first objectively interpolated over

the computation grid. The interpolated values provide the initial fields for a variational adjustment based upon conservation of mass, momentum, and flux of total energy in the layer. Figure 3 is a definition sketch of the model geometry. The model uses a dimensionless varable, σ, for vertical position in the layer, defined by,

$$\sigma = \frac{z - h}{D} \qquad 0 < \sigma < 1,$$

and model output consists of seven (7) fields, all given in mks units:

1. H (the height of layer top, MSL)

2. $\langle u' \rangle = D \int_0^1 \rho u(\sigma) d\sigma$ (x momentum flux)

3. $\langle v' \rangle = D \int_0^1 \rho v(\sigma) d\sigma$ (y momentum flux)

4. $\langle u' | 1 \rangle = D \int_0^1 \rho \sigma u(\sigma) d\sigma$ (vertically weighted x momentum flux)

5. $\langle v' | 1 \rangle = D \int_0^1 \rho \sigma v(\sigma) d\sigma$ (vertically weighted y momentum flux).

6. $\langle u'E \rangle = D \int_0^1 E(\sigma) u(\sigma) d\sigma$ (x energy flux)

7. $\langle v'E \rangle = D \int_0^1 E(\sigma) u(\sigma) d\sigma$ (y energy flux)

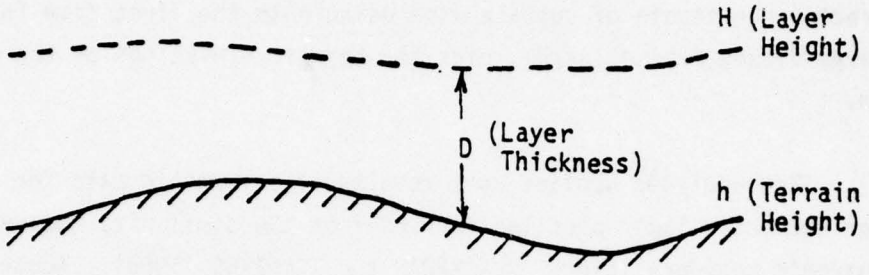


Figure 3 Layer Geometry of Mesomodel

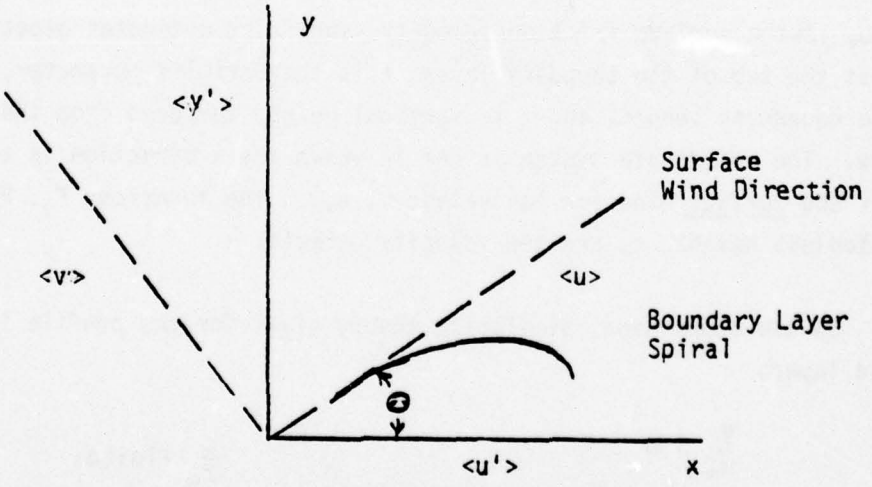


Figure 4 Coordinate Transformation

These data contain no direct information of wind and temperature profiles, only integrals of these quantities through the layer. No temperature information is obtained from the mesomodel output. The data analysis extracts an estimate of surface wind using only the first five (5) fields (energy fluxes are not used), plus the terrain elevations of the mesoscale grid.

The analysis applies some results of asymptotic matching of surface layer and outer layer profiles derived from the similarity theory of neutral barotropic boundary layers, BLACKADAR and TENNEKES (1968). According to these, in the outer part of the boundary layer the velocity profile is,

$$\frac{V-V_g}{u_*} = F_y(\zeta) \qquad \frac{z}{z_o} \to \infty$$

$$\frac{U-U_g}{u_*} = F_x(\zeta) \qquad \zeta = \frac{zf}{u_*} \text{ finite}$$

where u_* is the surface friction velocity, subscript g denotes geostrophic winds at the top of the boundary layer, f is the Coriolis parameter, z_o is surface roughness length, and z is vertical height measured from the ground surface. The coordinate system is one in which the x direction is the direction of the surface wind (or equivalently, u_*). The functions F_x, F_y of the dimensionless height, ζ, are the velocity defects.

On the other hand, similarity theory gives for the profile in the surface layer,

$$\frac{V}{u_*} = 0 \qquad \frac{z}{z_o} \text{ finite.}$$

$$\frac{U}{u_*} = f_x\left(\frac{z}{z_o}\right) \qquad \zeta = \frac{zf}{u_*} \to 0$$

one form of which is the logarithmic law.

The data analysis assumes that layer height of the mesoscale model is the boundary layer height. Convenient functional forms of the similarity laws are assumed which can match surface and outer layer profiles, and also the upper and lower boundary conditions. Specifically we assume,

$$\frac{U-U_g}{u_*} = \frac{1}{k} \ln \zeta + B$$

$$\frac{V-V_g}{u_*} = \frac{A}{k} (1-\zeta^2/\zeta_D^2) \qquad \text{outer layer}$$

$$\frac{U}{u_*} = \frac{1}{k} \ln \left(\frac{z}{z_o}\right) \qquad \text{surface layer} \quad z \lesssim 0.1\ D \qquad \text{(III.1)}$$

$$\frac{V}{u_*} = 0$$

where k is the von Kármán constant, and D is the layer thickness. The requirement that winds be geostrophic at the top of the boundary layer yields:

$$\frac{U-U_g}{u_*} = \frac{1}{k} \ln \left(\frac{z}{D}\right)$$

$$0.1 \lesssim z/D < 1 \quad \text{outer layer} \qquad \text{(II.2)}$$

$$\frac{V-V_g}{u_*} = \frac{A}{k} (1- z^2/D^2)$$

Matching of the surface and outer profiles at some arbitrary point near the top of the surface layer ($z/D \simeq 0.1$) gives the additional relations:

$$U_g = \frac{u_*}{k} \ln \left(\frac{D}{z_o}\right) \qquad \text{(III.3)}$$

$$V_g \cong - \frac{Au_*}{k}$$

Using Equation (III.3) the profile relations of Eq. (III.1) and (III.2) may be inserted into the definitions of the mesomodel output quantities. Performance of the integration gives formulas for the mesoscale output quantities in terms of the similarity parameters. As an example consider <u>.

$$\langle u \rangle = D \int_0^1 \rho u(\sigma) d\sigma$$

$$= D \int_0^{\sigma'} \rho \frac{u_*}{k} \ln \left(\sigma \frac{D}{z_o}\right) d\sigma + D \int_{\sigma'}^1 \rho \left[\frac{u^*}{k} \ln\sigma + \frac{u^*}{k} \ln\left(\frac{D}{z_o}\right) \right] d\sigma$$

The lower limit of integration is zero, instead of z_o/D, which introduces a very small error of order z_o/D. Partitioning of the integral into two parts given by the surface and outer layer profiles proves unnecessary since the integrands are the same (a result of the asymptotic matching). If one further assumes that density can be considered independent of height the result of this and similar integrations are the following formulas:

$$\langle u \rangle = \frac{D\rho u^*}{k} \left[\ln \left(\frac{D}{z_o} \right) - 1 \right]$$

$$\langle v \rangle = \frac{-D\rho u^* A}{3k}$$

$$\langle u|1 \rangle = \frac{D\rho u^*}{2k} \left[\ln \left(\frac{D}{z_o} \right) - 1 \right]$$

$$\langle v|1 \rangle = - \frac{D\rho A u^*}{4k}$$

These formulas, giving scaling parameters in terms of mesomodel output quantities, are still not in quite the correct form since they refer to a coordinate system aligned with the surface wind. Mesoscale output quantities are components in two perpendicular directions (x, y) which are rotated 45 degrees with respect to geographical north and east directions. The definition sketch of Figure 4 shows the coordinate rotation required. Relations of

similarity parameters to mesomodel output are then obtained.

$$\frac{D\rho u^*}{k}\left[\ln\left(\frac{D}{z_o}\right) - 1\right] = \langle u'\rangle \cos\Theta + \langle v'\rangle \sin\Theta$$

$$-\frac{D\rho u^* A}{3k} = -\langle u'\rangle \sin\Theta + \langle v'\rangle \cos\Theta$$

$$\frac{D\rho u^*}{2k}\left[\ln\left(\frac{D}{z_o}\right) - 1\right] = \langle u'|1\rangle \cos\Theta + \langle v'|1\rangle \sin\Theta$$

$$-\frac{D\rho A u^*}{4k} = -\langle u'|1\rangle \sin\Theta + \langle v'|1\rangle \cos\Theta$$

Knowing the layer thickness, D, and the roughness length, z_o, this set of equations is solvable for the four quantities, u^*/k, A, $\sin\Theta$, $\cos\Theta$.

Consistent with the assumption of a neutral, barotropic boundary layer (the only assumption possible with the given data) the surface layer wind field is then determined by,

$$u = \frac{u^*}{k} \ln\left(\frac{z}{z_o}\right) \cos\Theta$$

$$v = \frac{u^*}{k} \ln\left(\frac{z}{z_o}\right) \sin\Theta$$

where z is the user selected computation height. These components, oriented in the coordinate system of the mesoscale model, are then resolved into geographical east and north components.

These theoretical formulas are implemented in the program segment MESVAR to obtain an estimate of surface background wind from the mesomodel output data obtained by the program segment MESMOD.

III.2 SPATIAL AND TEMPORAL EXTRAPOLATION OF UPPER AIR AND NUMERICAL PREDICTION DATA

Upper air profiles derived from Rawinsonde observations are available

in the data base at station locations distributed throughout the southwestern U.S. The spatial density of these observations is low. The closest observation may be of the order of 100 km away from the site of application of the microscale terrain wind model. Also, the observation times may be as much as 12 hours previous to the simulation time. In addition, the data base contains rough profiles of upper air quantities generated by the USAF Global Weather Central "fine-mesh" numerical predictions. These quantities are located on a grid of approximately 150 km, and are available at hourly time intervals. (Predictions extend for 18 hours beginning at the analysis times of 00Z and 12Z. In real time operation the first six hours of each prediction sequence would be unavailable.) This section describes the analysis procedure used to spatially interpolate and update these profiles in order to obtain estimates of potential temperature profiles and, where required by the wind estimate, standard pressure heights and their gradients at the simulation site. Presently the analysis is designed to interpolate the standard pressure heights (850, 700, and 500 mb only) and the potential temperatures at these heights. Winds at these standard pressure levels could also be interpolated.

In the following we describe the analysis of the southwest U.S. upper air data. Analysis of the European data base employs the same general methods, with some differences which are explained at the end of the section.

The basic procedure operates by the establishment of planes for linear interpolation. As an example consider the 700 mb pressure height. At a given time, t_1, this pressure height can be approximated over the local region near the simulation site by a plane,

$$z_{UA,700}(t_1) = a_U(t_1) + b_U(t_1)x + c_U(t_1)y$$

where the coefficients depend upon the type of data and the time. Let us further assume that this plane interpolates upper air observations at time t_1. The coefficients are found from pressure heights at the station locations. The data search finds the four (4) or less closest stations within a fixed

distance of the simulation site. If three (3) stations are present the coefficients are exactly determined. If four (4) stations are found the coefficients are over determined, a case handled by determining the best least squares fit of a plane to the four (4) data points. The case of only one (1) or two (2) stations requires supplementary information provided by the GWC data. The GWC prediction data at the four (4) closest grid points to the simulation site (or as sometimes occurs, only three) at the same time, t_1, also determines a plane interpolating the GWC 700 mb height.

$$z_{GWC,700} = a_G\,(t_1) + b_G\,(t_1)x + C_G\,(t_1)y$$

In the case of only one (1) upper air observation the slopes of the 700 mb pressure height provided by the GWC plane are used, i.e.;

$$b_u(t_1) = b_G(t_1)$$

$$c_u(t_1) = c_G(t_1).$$

while the constant $a_u(t_1)$ is provided by the single upper air observation.

Two (2) upper air observations suffice to determine the intersection of the interpolative plane with a vertical plane through the two stations. The slope of the interpolative plane normal to this intersection is established by using the slope in this direction of the GWC plane. Computationally the case of two (2) upper air observations is more complex than the others, but follows from standard analytical geometry.

The foregoing example illustrates the establishment of an interpolative plane for one item of upper air data, 700 mb height, from which the 700 mb height, and its slope, may be determined at time t_1, the time of the station observations, at any point in a local region. Real data, the observations, are used to the maximum extent possible; but the presumably lower grade GWC predictions are incorporated if observational data is incomplete. Note that

only slopes, not magnitudes, of GWC interpolative planes are introduced. Provided a single upper air observation is present, the interpolative planes are normalized with it. GWC data is used to establish only spatial trends (i.e., slopes) which cannot be determined otherwise. Of course, if observational data is entirely absent, the GWC prediction data is used alone as a default.

If the time selected for the simulation coincides closely enough (<1 hr.) with the time of upper air observations, the procedure illustrated above is applied to establish interpolative planes of upper air quantities. If, as is usually the case, the upper air observations are one hour or more old relative to simulation time and GWC predictions are available, GWC data are used to establish updated extrapolative planes of upper air data. Suppose the most recent observation time is t_1 and the simulation time is t_2, where t_2-t, $\geq$1 hr. Again, take the 700 mb pressure height as an example. The GWC data provide interpolative planes at each of two times.

$$z_{G,700}\ (t_1) = a_G(t_1) + b_G(t_1)\ x + c_G(t_1)y$$

$$z_{G,700}\ (t_2) = a_G(t_2) + b_G(t_2)\ x + c_G(t_2)y$$

The spatial interpolation of the temporal trend of z_{700} in the GWC predictions is given by the difference of these equations.

$$\Delta z_{G,700} = [a_G(t_2) - a_G(t_1)] + [b_G(t_2) - b_G(t_1)]\ x$$

$$+ [C_G(t_2) - c_G(t_1)]\ y$$

The available upper air observations are updated by adding the GWC trend for the time interval, evaluated at the location of the observation site, (x_i, y_i).

$$z_{i,U,700}(t_2) = a_{i,U,700}(t_1) + \Delta z_{G,700}(x_i, y_i)$$

The updated observations are then used to establish interpolation planes of

upper air observations valid at simulation time. Depending upon the number of observation stations GWC predictions may still be required, as above, to completely establish the interpolation planes.

The analysis of upper air profiles is founded upon the philosophy of using real data, if available, in conjunction with spatial and temporal trends derived from GWC numerical predictions. Although absolute magnitudes of GWC predictions may be inaccurate, these numerical predictions take large scale synoptic changes into account. Spatial and temporal trends, but not magnitudes, are the information contained within the GWC data which is deemed most valuable. Our analysis incorporates these notions in a simple, straightforward manner, but by no means exhausts the possible analysis techniques.

The theoretical approach sketched here is implemented in the data analysis program to provide upper air profiles at simulation time of pressure heights and potential temperatures not only at the simulation site itself, but also at surface station locations. The profiles at surface station locations, as described in another section, are central for interpretation of surface observations to provide input parameters of surface temperature and general wind to the microscale model. One computational aspect of the establishment of interpolation planes is of interest. Since the locations of upper air observing stations, or GWC grid points, do not change; computational economy is achieved by constructing the planes from linear combinations of "basis" planes. A basis plane (or linear function) is associated with each data location and has unit value at that location. Its value at the other station locations is zero. In the case of overdetermined planes (4 data locations) the values are as close to one and zero as a least squares fit permits. Denoting the basis function associated with each data location, i, by $l_i(x, y)$ the interpolation plane is given by,

$$z(x,y) = \sum_{i=1}^{N} z_i \, l_i \, (x,y)$$

where z_i is the value of the interpolated quantity at each of the N data locations.

The European data base contains no numerical predictions, so an alternative procedure is provided to establish interpolative planes of upper air data at simulation time. By setting a flag, these planes can be determined by linear interpolation in time between spatial extrapolative planes of upper air data at two times, past and future, which bracket the simulation time. This flag needs to be set for the analysis of the European data, or any other case in which numerical prediction data is missing. If the flag is not set and predictive data are not acquired, upper air data less than 3 hours prior to simulation are used as a last resort. If the only upper air data available are older than 3 hours, no simulation time interpolative planes of upper air data are generated.

The updated planes for the spatial extrapolation of upper air data are established by the program segment PROFLS.

III.3 ANALYSIS OF SURFACE STATION DATA

The observational data from surface stations, although they may be located up to 100 km away from the simulation site, is still of importance in determining input parameters to the local surface wind analysis. Their primary value is timeliness. The data base contains in some cases hourly observations, and observations taken less than three (3) hours previous to simulation time at stations within 100 km of the simulation site are generally available.

The general data analysis may use the surface data for two purposes. It is the sole source of current surface air temperature information. Surface wind observations may also be used to calibrate and normalize an estimate of simulation site wind derived via the geostrophic approximation.

The analysis of surface temperature data makes the basic assumption that ΔQ, the surface heating at the observation sites, is representative of the surface heating at the simulation site as well. At each surface station location the potential temperature profile, derived from the upper air data analysis, is extrapolated to ground level to obtain a "background" surface

potential temperature, Θ_o. The "actual" surface potential temperature Θ_s is computed from the reported surface temperature T_s and the pressure given by an extrapolation to ground level of the pressure height profiles. From the definition of potential temperature and the thermodynamic TdS equation -- assuming heating occurs at constant pressure -- the heat change at the surface station relative to the "background" temperature is:

$$\frac{\Delta Q}{C_p} = \frac{T_s(\Theta_s - \Theta_o)}{\Theta_o} \quad .$$

This quantity averaged over the surface stations is taken to represent surface air heating at the simulation site relative to the background temperature obtained by extrapolation to the surface of upper air profiles.

The potential temperature estimate at each grid point in the simulation area is obtained by solving the above formula for Θ_s,

$$\Theta_s = \Theta_o + \left(\frac{1000}{P_o}\right)^{R/C_p} \left\langle \frac{\Delta Q}{C_p} \right\rangle$$

where < > represents the average of surface observations, and the zero subscripts now denote extrapolations to ground level of upper air profiles at points in the simulation area. In obtaining this formula the approximation,

$$T_s \cong \Theta_o \left(\frac{P}{1000}\right)^{R/C_p}$$

is employed. The field of surface potential temperature, thus established, exhibits structure dependent upon terrain height variations within the simulation zone.

Use of the surface station wind data first requires a computation of

the surface geostrophic wind at each site. In sigma coordinates the surface geostrophic wind ($\sigma = 1$) is given by,

$$0 = - (g \nabla z_s + RT_s \nabla \ln p_s) - f \hat{k} \times \vec{V}_g$$

where the subscript, s, refers to surface values, $\hat{k}$ is a vertical unit vector, and z is terrain height. Other symbols have their usual meteorological meaning. DANARD (1977) shows that a simplification of the expression in parentheses consistent with the assumption of linear vertical variation of pressure force is given by:

$$g \nabla z_s + RT_s \nabla \ln p_s = g \nabla_p z_{85} - \vec{n} (z_{85} - z_s)$$

where,

$$\vec{n} = g \nabla_p \ln T_{85}.$$

The subscript, 85, refers to the 850 mb level, but for elevated terrain the first standard pressure level above the terrain surface may be used equally well. The horizontal gradients at constant pressure levels and the level heights themselves are given directly by the interpolation planes of upper air data. These formulas permit calculation of the surface geostrophic wind at each surface station location.

The somewhat fictitious surface geostrophic wind does not account for advective, frictional, diffusive, and time dependent effects. Consequently measured surface winds vary widely from it. However its dependence solely upon upper air conditions and terrain heights renders the geostrophic wind the best vehicle, short of a mesoscale model, to carry the extrapolation/ interpolation of surface measurements to the simulation site. We use it for this purpose.

At each surface station the difference, both in direction and magnitude, between the measured wind and the geostrophic wind is computed. Then a test is made of the statistical scatter of these differences and the magnitude of their averages. If the criteria of the test are satisfied, the average value of the differences is applied to the geostrophic wind at the simulation site to obtain an estimate of the surface level general background wind there. Establishment of reliable criteria for these tests will require some exercise with the data bases. Presently reasonable, but generous, criteria are used; i.e., directional relative variance < 50%, and magnitude relative variance < 50%. If the criteria are not met, and failing other information, the magnitude and direction of the wind at the closest surface station is used as a default.

The general wind estimate produced by this analysis technique cannot reflect local topographic influence upon the wind field (except the effect of surface elevation), but the local surface analysis routine itself may yield these. Frictional effects and modifications of the surface wind field attributable to boundary layer thickness and stability are assumed to be similar at both the surface stations and the simulation site.

A more elaborate data analysis procedure was considered, one using the boundary layer parameterizations based upon general similarity theory as given by ARYA (1977). These parameterizations employ both the boundary layer height and the Obukhov stability length as well as the friction velocity and the geostrophic wind. The surface station data together with the upper air profiles in principle contain enough information to determine the required parameters. But the problem of spatially extrapolating the boundary layer height still remains. Consequently the more elaborate, and physically more satisfying, data analysis procedure was discarded in favor of the empirical procedure described above.

The analysis of surface station data is implemented in two program segments. The routine SRFTMP uses surface data to estimate surface heating at the simulation site, and the routine UAVAR performs the calculations

required for estimating the general background wind at the simulation site via the geostrophic extrapolation.

IV. GENERAL PROGRAM STRUCTURE

The block diagram of Figure 5 displays the overall structural logic of the numerical analysis program to integrate the high resolution wind analysis as a part of EPAMS. Segmentation of major program elements and levels is indicated. XECAMS, the overall executive, determines that an analysis routine is to be run, and passes control to ANALYZ. The supervisor ANALYZ, on the basis of flags received from the executive, determines which particular analysis is required, and in this case passes control to WNDMGR (Wind Manager) which controls all functions required for the surface layer wind analysis. Subordinate to WNDMGR are the major program segments TERACQ (Terrain Acquisition), DTBMGR (Data Base Manager), DATANL (Data Analysis), DRIVRS (Driving Parameters), and WINDEX (Wind Extrapolation). The major routines are executed sequentially as specified by control flags supplied by namelist input. The general functions of each of these routines is as follows:

WNDMGR acquires directory information from the micro terrain and wind files, reads input specific to the wind model, assigns default values, computes Julian times, and updates the micro wind directory upon successful execution of the calculation. As the overall manager, it calls and monitors the major subordinate program elements.

TERACQ obtains the pre-processed micro terrain elevation and surface roughness data fields for the area closest to or at the user specified UTM coordinate location of the local simulation. If no terrain data sufficiently close to the simulation location exists, a message is printed and execution of WINDEX is suppressed.

DTBMGR scans the files of meteorological data in the EPAMS data base and extracts relevant data. Any combination of data types or none at all, depending upon user set flags, can be extracted.

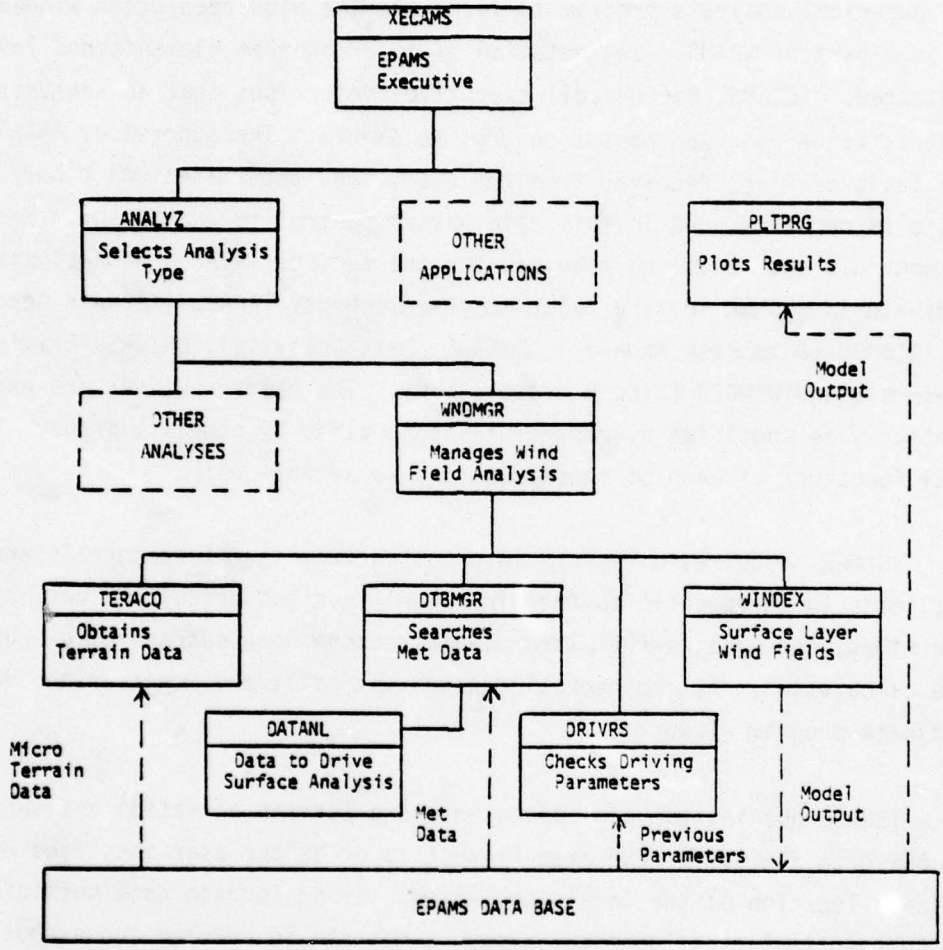


Figure 5. Structure of Surface Wind Analysis in EPAMS

DATANL uses the selected data obtained by DTBMGR to generate the meteorological parameters required to drive the surface layer analysis. These parameters; background wind at the local simulation area, a vertical profile of potential temperature and an estimate of surface heating; are obtained by use of the analysis described in Section III. Owing to the varying types, relevance, and availability of meteorological data in the data base, this program segment exhibits some logical complexity based upon priorities of data usage or user specified options. It also permits direct user input of driving parameters in lieu of driving parameters computed from the data base.

DRIVRS compares the driving parameters determined by DATANL with driving parameters used to generate previous high resolution wind analyses over the same terrain block. If the driving parameters nearly match those used to generate windfields archived in the microwind file, an informative message to that effect is printed and execution of WINDEX is suppressed.

WINDEX is the program segment which produces the final high-resolution surface wind field. This segment performs the core of the numerical surface layer analysis by a relaxational adjustment of wind and temperature fields as described in Section II. At the conclusion of the relaxation WINDEX outputs the estimated high resolution wind and temperature fields to the microwind files of the EPAMS data base.

Directly below the EPAMS executive in Figure 5 is shown a PLTPRG (Plot Program) segment. This independent utility program is executed separately from the other program structure to provide plots of terrain elevation in the simulation area or wind and temperature fields generated by the high resolution wind analysis.

Another independent utility program TERPRO (Terrain Processor), which is not shown in Figure 5, has been constructed to produce suitable blocks of micro terrain data for use by the surface layer analysis from UTM topographic data tapes.

Following this overview further details of the operation of the major program segments are given in the following sections.

V. MAJOR PROGRAM SEGMENTS

The function and internal organization of the major program segments, directly or indirectly subordinate to WNDMGR are described in more detail.

V.1 DTBMGR (Data Base Manager)

The organization of DTBMGR is shown schematically in Figure 6, which indicates the types of meteorological data which may be sought. This search includes substantially all the data types available in the EPAMS data base. The criteria for usable types of data depend upon the considerations explained under DATANL below. The function of DTBMGR is to obtain from the data base the data which is potentially relevant, and to store it in a readily accessible memory block for subsequent processing by DATANL. Each of the routines subordinate to DTBMGR accesses a particular type of data. Through user specified flags any combination of data types, including all or none, are accessed. If user flags are not set, DTBMGR accesses all available data types. In DTBMGR criteria of data relevance are based simply upon the time and location of the surface-layer analysis to be performed.

MRCUA seeks the most recent upper air sounding data (up to 12 hours old) from observation stations within 250 km. of the simulation site. The inventory of upper air data is read until data less than 12 hours old is found. A list of upper air stations within 250 km. of the simulation site, ordered by increasing distance, is generated. Upper air data from these stations is read until four stations have been obtained or the list is exhausted. An option provides for acquisition of upper air data at two time levels, within 12 hours after and before simulation time, for interpolative purposes when GWC prediction data is either not present or not desired.

MRCSFC seeks data from surface observations at times previous to simulation time up to a time limit, in hours, set by input flags. The search

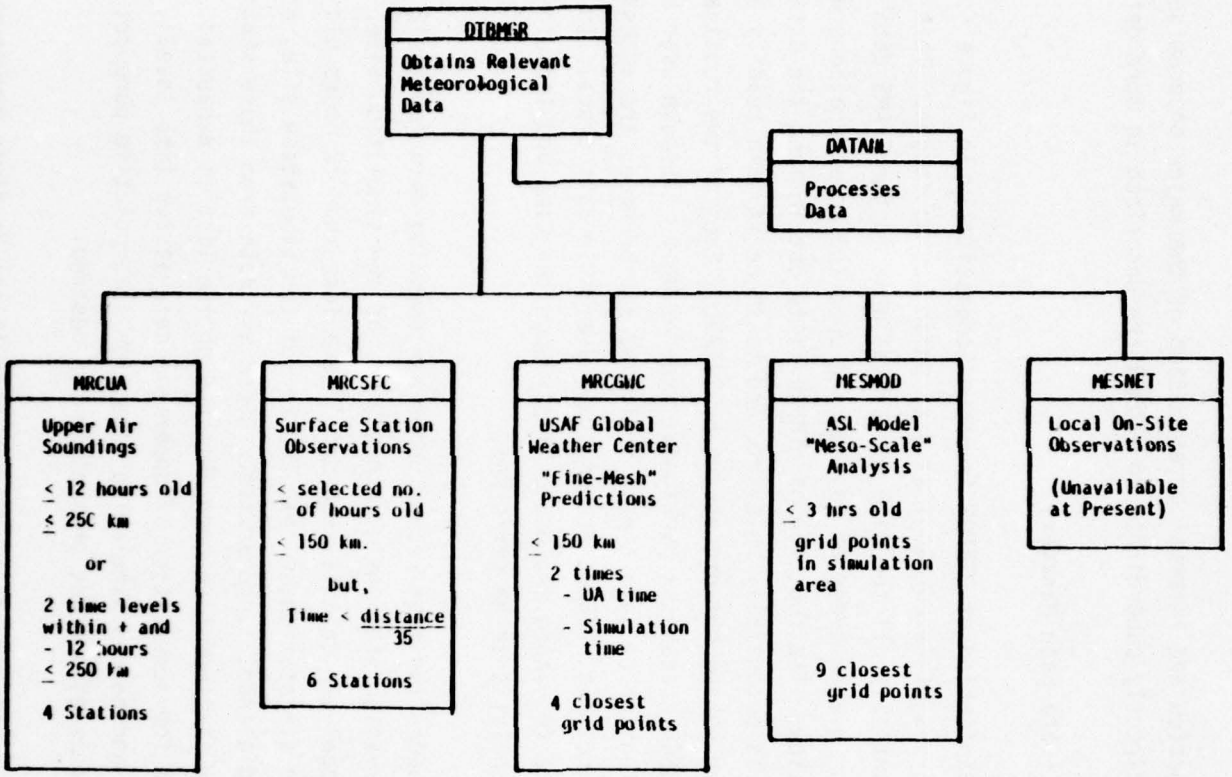


Figure 6. Structure of Data Base Manager (DTBMGR)

extends to 150 km from the simulation site, but this spatial range is decreased by 35 km for each hour of time difference between the simulation time and the time of the observation. The search procedure produces a list of observations which satisfy these criteria. Data from this list is read until six data sets are found or the list is exhausted.

MRCGWC seeks USAF Global Weather Central "Fine-Mesh" prediction data at the four GWC gridpoints closest to the simulation area. Data is sought at two time levels, the actual simulation time and the time at which the previously found upper air sounding observations were made. (In the case of synoptic soundings, the latter time is the GWC analysis time.) The search procedure reads the GWC inventory, creates a list of grid points closest to the simulation site, and determines the block containing the GWC data at the time required. The time is first set equal to the upper air observation time, and GWC data sets are read until four data sets are found or the list is exhausted. The procedure is then repeated for the simulation time.

MESMOD acquires data previously generated by the ASL meso-scale model at the nine grid-points (grid interval, 5 km) closest to or in the simulation area. These data comprise the two components of wind momentum averaged through a layer, the two vertical moments of wind momentum averaged through the layer, the two components of total energy flux averaged through the layer, and the variable layer height. These data, resulting from a diagnostic meso-scale analysis, are deemed applicable if their analysis time is within three hours of the prescribed simulation time.

MESNET is at present a dummy routine, inserted to enable future consideration of data acquired from a local mesoscale observational network.

The accessing of upper air, surface, and GWC data in the foregoing routines is a substantial modification of routines previously used for data accessing by DUMBAULD and BJORKLUND (1976).

Placing of the major program segment DATANL subordinate to DTBMGR prevents the large block of data acquired by DTBMGR from appearing at a higher program level.

V.2 DATANL (Data Analysis)

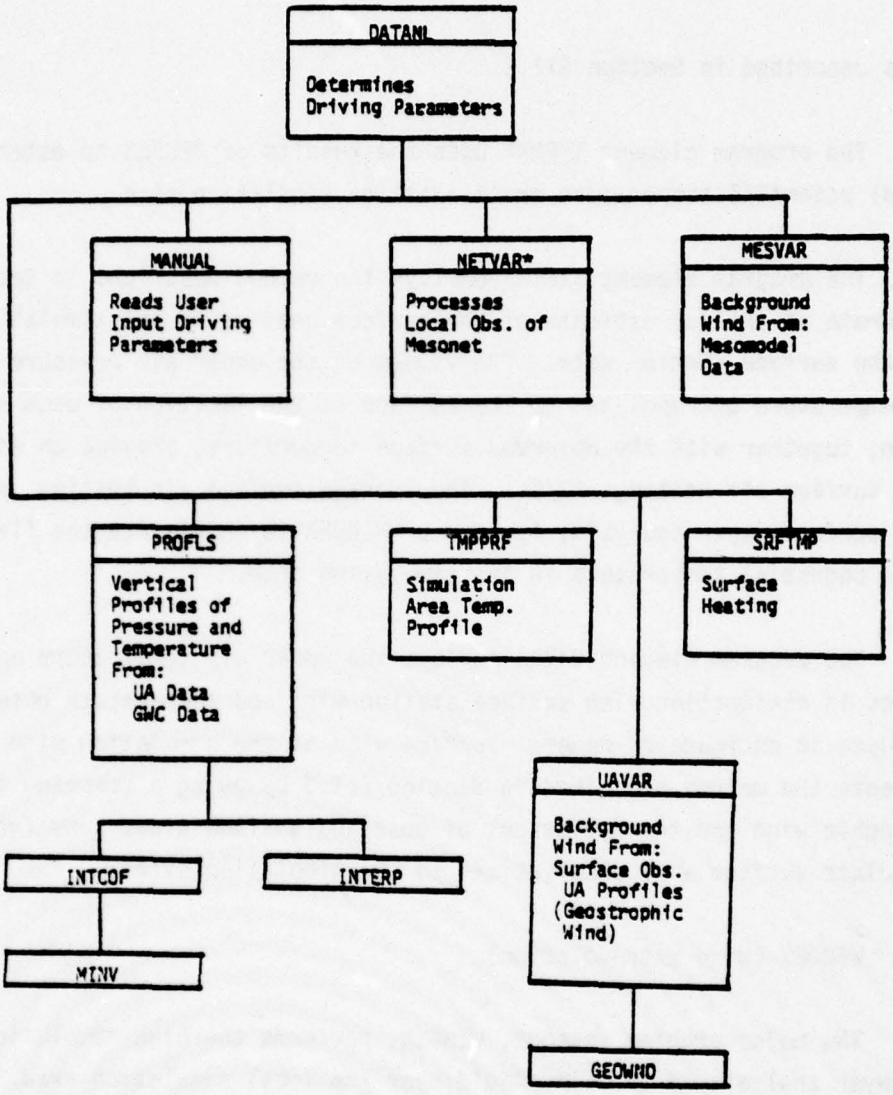
Though subordinate to DTBMGR, data analysis constitutes a major program segment. The function of the data analysis routine (DATANL) is to generate from the relevant data extracted from the data base by DTBMGR the parameters required to drive the surface layer wind analysis, WINDEX, as described in Section III. The subordinate elements of DATANL are shown by Figure 7. DATANL executes these elements as determined by priority ordering and data availability or as specified by user supplied input flags.

The program element MANUAL reads user supplied driving parameters for WINDEX when they are present and their use is specified.

The program element NETVAR is an inoperative dummy routine, included to provide a slot for future use of local observational data from within the simulation area to generate input parameters for the surface wind analysis.

The program element MESVAR extracts a value of surface wind in the simulation area from data previously generated by the one-layer ASL mesoscale model. The methods described in Section III.1, employing asymptotic matching of surface layer and outer layer profiles, BLACKADAR and TENNEKES (1968), are used to establish a surface layer wind estimate from the mesoscale model output.

The program element PROFLS, together with its subordinate routines as indicated in Figure 7, determines the interpolative planes of upper air data over a large geographic area which includes the simulation area. These interpolative planes, valid at the simulation time, are established by the



*A dummy, not presently used.

Figure 7. Elements of Data Analysis (DATANL) Segment

methods described in Section III.2.

The program element TMPPRF uses the results of PROFLS to establish th vertical potential temperature profile at the simulation site.

The program element SRFTMP employs the method described in Section III.3 to generate an initial estimate of the surface heating in the simulation area using the surface station data. The values of the upper air pressure and potential temperature extrapolated to the surface at the location of each surface station, together with the observed surface temperature, provide an estimate of the surface air heating, $\Delta Q/C_p$. The average surface air heating, an input to the surface layer analysis, is used by WINDEX to initialize the field of surface potential temperature in the simulation area.

The program element UAVAR employs the upper air temperature and pressure profiles in conjunction with surface station wind and temperature observations to produce an estimate of general surface wind at the simulation site. UAVAR implements the method described in Section III.3 by using a computed surface geostrophic wind and the deviations of observed surface winds away from it to extrapolate surface wind observations to the simulation site.

V.3 WINDEX (Wind Extrapolation)

The major program segment, WINDEX, performs the high resolution surface layer analysis of the wind field over the local simulation area. It accepts the data supplied by DATANL consisting of an estimated uniform surface wind over the simulation area, the vertical profiles of potential temperature and pressure at the simulation site, and an estimate of surface air heating. In addition to these metorological parameters WINDEX uses high resolution arrays of terrain elevation and surface roughness. The organization and subordination of the constituent elements of WINDEX are shown in Figure 8.

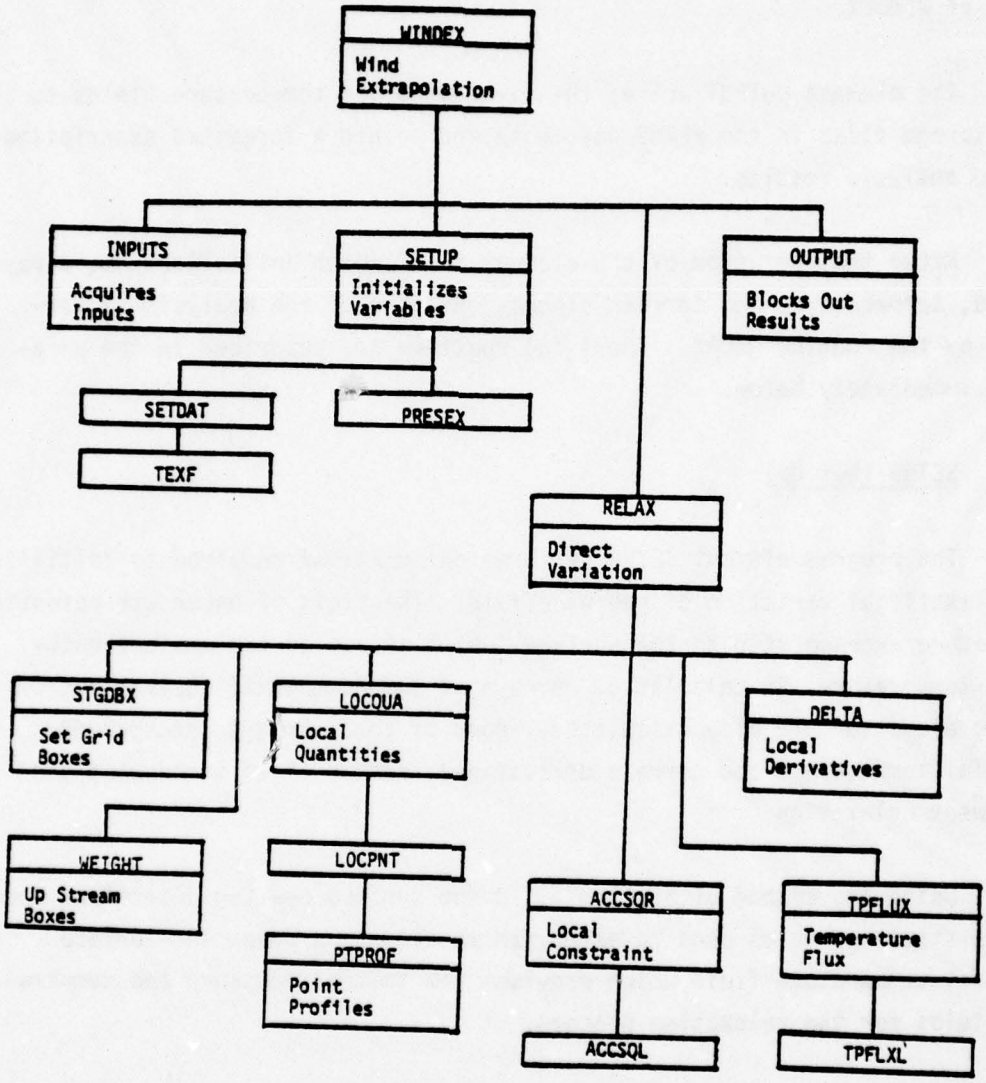


Figure 8. Subordinate Elements of WINDEX

The element INPUTS initializes constants and transfers data to input arrays of WINDEX.

The element OUTPUT writes the final wind and temperature fields to mass storage files in the EPAMS data base and prints a formatted description of wind analysis results.

After the operation of the element SETUP which initializes the arrays of wind, temperature, and terrain slopes, the core of the analysis is performed by the routine RELAX. These two routines are described in the paragraphs immediately below.

V.3.1 SETUP (Set Up)

The program element SETUP performs calculations required to initialize the relaxational variation of the windfield. The field of upper air potential temperature extrapolated to the surface, which serves as the ambient background temperature, is calculated. Arrays of the horizontal derivatives of terrain elevation are also calculated. Both of these arrays, background potential temperature and terrain derivatives, remain fixed throughout the subsequent relaxation.

Using the method of Section III.3 the surface heating determined from surface station data is used to establish an elevation dependent surface potential temperature field which provides the initial buoyancy and temperature fields for the relaxation process.

The arrays of surface wind field components are initialized to the uniform general wind provided by DATANL. A coordinate rotation is required since internally WINDEX uses wind components rotated 45 degrees with respect to the terrain axes. (See Appendix B and Figure B.1.) After completion of the calculation, OUTPUT inverts this rotation to provide output wind components.

V.3.2 RELAX (Least Constraints Relaxation)

The program element RELAX implements the concepts of Section II by a set of numerical algorithms which perform a variational relaxation of wind and temperature fields in the surface layer to generate a physically based estimate consistent with mass conservation, the equation of motion, surface boundary conditions, the constraint of the convoluted terrain surface, and available meteorological information.

RELAX performs a user prescribed number of relaxation sweeps over the grided arrays of velocity and temperature fields. During a single sweep corrections of velocity and temperature to reduce the integrals of the model equations (II.11) are computed at each grid point and stored. During the sweep the local contributions to each of the integrals are summed to provide the total value of each of the integrals at the conclusion of the sweep. After the sweep is completed the corrections of temperature and velocity are applied at all grid points, a simultaneous relaxation of the fields. RELAX contains two sets of arrays of wind and temperature. In addition to the wind and temperature arrays which are being relaxed, the wind and temperature arrays corresponding to the minimum constraint obtained up to the current stage of the relaxation are saved. When the prescribed number of relaxation steps is completed these fields, corresponding to the minimum constraint at any stage of the relaxation, are output as the best estimate.

Calculation of local contributions to the dynamic constraint and temperature flux integrals requires that all subordinate routines of RELAX be called at least once, and sometimes several times, at each grid point during each relaxation sweep. As detailed in Appendix B, the local contributions to the integrals are calculated in terms of fluxes out of volume elements which subdivide the computational layer. A given grid point at which velocity components and temperature are defined can appear on the surface normal face of each of four overlapping flux boxes. The flux boxes couple a field quantity at one grid point to the field quantities at each of the surrounding eight

grid points. Therefore a local grid consisting of these nine grid points is provided in the structure.

The routine STGDBX (Set Grid Boxes) determines whether the current grid point is in the interior, on the boundary, or in the corner of the array and sets the box indices which determine which of the surrounding four flux boxes are possible. In the interior all four are possible, at the boundary only two, and in a corner only one.

The routine WEIGHT zeros the box indices of all but the two flux boxes most nearly upstream of the current grid point, so only upstream flux boxes are used. (A user option provides for use of all 4 flux boxes, if desired.) At boundaries and corners possible flux boxes are used regardless of wind direction.

The routine LOCQUA (Local Quantities) sets up the correspondence between indices of the local 9-point template and the global grid indices of the simulation area. Through calls to its subordinate routines, LOCPNT (Local Point) and PTPROF (Point Profiles), it acquires or computes the values of all variables needed to evaluate the contributions of the local flux boxes to the integrals of constraint and temperature flux divergence. The selection of flux boxes made previously in WEIGHT determines at which points in the local 9-point template these quantities are required. Because typically two flux boxes are used, the local 9-point template is not completely filled. Each filled point of the local template contains: the two horizontal derivatives of terrain elevation, an area scaling factor of sloping terrain, the exponent of the power law wind profile, a momentum flux averaging factor of the power law profile, the surface normal derivative of potential temperature, the buoyancy, two components of wind velocity, and the local temperature. This set of routines uses the methods described in Section II.4.3 to determine the wind and temperature profiles at each point.

The segment ACCSQR (Acceleration Squared, Resultant) computes the least constraint integral for each of the local flux boxes used, as well as the

surface normal acceleration in each of the flux boxes. (The surface normal acceleration is used in the next relaxation sweep by the routine PTPROF to compute flow curvature corrections to the stability). The results for the flux boxes are then averaged.

The actual calculation of the constraint integral of a single flux box is done by the routine ACCSQL (Acceleration Squared, Local). This routine employs the algorithms derived in Appendix B based upon the terrain following coordinate system described in Section II.4.2.

The routines TPFLUX (Temperature Flux) and TPFLXL (Temperature Flux, Local) are structurally analogous to ACCSQR AND ACCSQL, respectively. They employ the algorithms of Appendix B to calculate the local contribution to the potential temperature flux integral.

To implement the relaxation scheme described in Section II.5 the segment DELTA calculates the derivative of the local constraint integral with respect to each of the two components of wind velocity at the simulation grid point. It also calculates the derivative of the local contribution to the temperature flux integral with respect to the local grid temperature. These derivatives are established through additional calls to ACCSQR and TPFLUX with slightly altered velocity and temperature.

The control routine RELAX uses these derivatives, accumulated during the relaxation sweep, to compute and apply relaxation corrections to the wind and temperature fields according to the method of Section II.5 prior to the initiation of the next relaxation sweep.

V.4 DRIVRS (Driving Parameter Test)

The major program segment DRIVRS (Drivers) performs the important executive function prior to the execution of WINDEX of testing whether the driving (input) parameters of WINDEX differ sufficiently from the driving

parameters of previous archived executions to justify a new execution of WINDEX. The driving parameters tested are the terrain data, background wind, and surface heating. The tests are the following: terrain data must be identical; the background wind estimates must agree to within 20% in magnitude and to within 10 degrees in direction; the surface heating estimates, $\Delta Q/C_p$, must agree to within 1.5 degrees. If all the tests are met for some archived set, the run number of the archived run is printed and WINDEX is not executed. If no archived set meets these tests, the surface wind file header is updated with the new driving parameters and WINDEX is executed.

surface normal acceleration in each of the flux boxes. (The surface normal acceleration is used in the next relaxation sweep by the routine PTPROF to compute flow curvature corrections to the stability). The results for the flux boxes are then averaged.

The actual calculation of the constraint integral of a single flux box is done by the routine ACCSQL (Acceleration Squared, Local). This routine employs the algorithms derived in Appendix B based upon the terrain following coordinate system described in Section II.4.2.

The routines TPFLUX (Temperature Flux) and TPFLXL (Temperature Flux, Local) are structurally analogous to ACCSQR AND ACCSQL, respectively. They employ the algorithms of Appendix B to calculate the local contribution to the potential temperature flux integral.

To implement the relaxation scheme described in Section II.5 the segment DELTA calculates the derivative of the local constraint integral with respect to each of the two components of wind velocity at the simulation grid point. It also calculates the derivative of the local contribution to the temperature flux integral with respect to the local grid temperature. These derivatives are established through additional calls to ACCSQR and TPFLUX with slightly altered velocity and temperature.

The control routine RELAX uses these derivatives, accumulated during the relaxation sweep, to compute and apply relaxation corrections to the wind and temperature fields according to the method of Section II.5 prior to the initiation of the next relaxation sweep.

V.4 DRIVRS (Driving Parameter Test)

The major program segment DRIVRS (Drivers) performs the important executive function prior to the execution of WINDEX of testing whether the driving (input) parameters of WINDEX differ sufficiently from the driving

VI INDEPENDENT UTILITY PROGRAMS

The surface layer analysis routine of EPAMS is complemented by some utility routines to facilitate its operation. These routines serve the functions of creating appropriate arrays of terrain data for the surface layer analysis and of providing a flexible plotting capability for surface layer analysis results. The terrain processing routines access files of topographic data available at the ASL and produce the blocks of high-resolution terrain data used by the surface layer analysis. A plotting routine is provided for graphic display of the surface layer analysis results which are stored in the EPAMS surface wind file. Three simple interactive routines are also supplied to aid in file management. The terrain processing and the plotting routines, while presently independent of the wind analysis, could readily be made a part of the EPAMS.

VI.1 TERRAIN PROCESSING (TERPRO)

Two programs perform terrain processing, one for the southwestern U.S. and one for a block of terrain in Europe (near Fulda, Germany).

Figure 9 shows the general structure of the SWUS program. Its function is to use the micro-terrain data available at ASL (consisting of some 400 80-by-80 blocks of terrain elevation data on the UTM grid at approximately 63.5 meter horizontal resolution) to create 40-by-40 blocks of terrain elevation and surface roughness length data usable by the surface layer analysis. BLKDRV acquires header data, reads input, and writes out updated header data.

BLKMGR calls its major subordinates, monitors their results, updates the header after each new pair of terrain elevation and surface roughness arrays has been created, generates roughness estimates based upon terrain elevation, and blocks out the terrain and roughness arrays to mass storage.

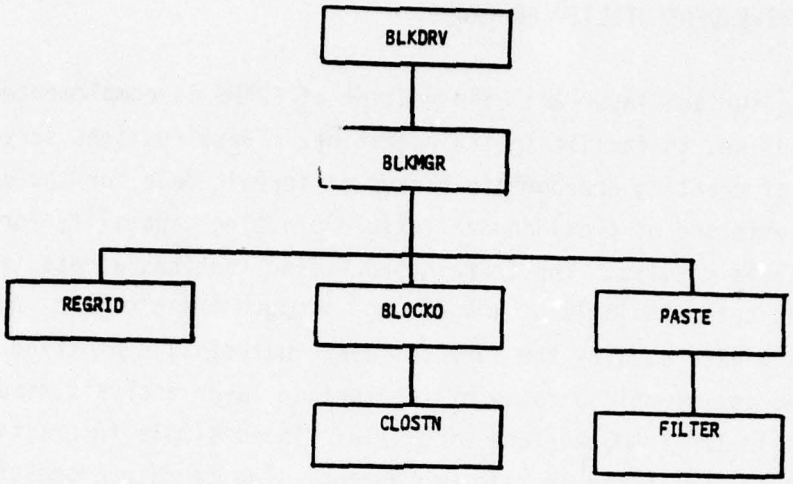


Figure 9. SWUS Terrain Processor (TERPRO)

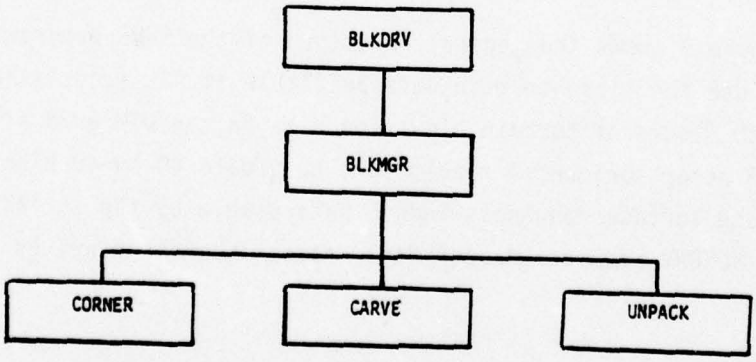


Figure 10. European Terrain Processor (TERPRO)

REGRID determines the grid spacing, dimensions, and UTM coordinates of the southwest corner of the block to be created. BLOCKO determines which blocks from the large input terrain file are needed to create the output block. PASTE keeps track of indices while pasting together the portions of already processed data which will compose the output block. FILTER acquires one of the input 80-by-80 blocks, averages terrain elevations, applies corrections to avoid excessive smoothing, and returns the portion required in the final output block.

Figure 10 shows the general structure of the program which processes the European terrain data. The initial European data file consists of a single 101-by-51 block of packed data which contains elevation and height of ground cover. The processing routine outputs a pair of 51-by-31 data blocks containing terrain elevation and surface roughness length.

BLKDRV and BLKMGR perform the same functions as the routines of the same name in the SWUS terrain processor. CORNER determines grid indices in the 101-by-51 block corresponding to UTM coordinates specifying the corners of the terrain block to be created. CARVE transfers the required 51 by 31 block from the larger block to a working array. UNPACK unpacks the data in the working array, computes roughness lengths from terrain cover heights, adds the displacement height of terrain cover to the terrain elevation, and returns the two blocks, terrain elevation and roughness length, to be stored in the output file.

VI.2 PLOTTING ROUTINES

Figure 11 displays the general structure of the MRC plotting program constructed to plot data generated by the surface layer wind analysis. The plotting program employs many program elements developed by ASL personnel and available on program files at ASL.

XECAMT and PLTPRG provide ready interfacing with the EPAMS control

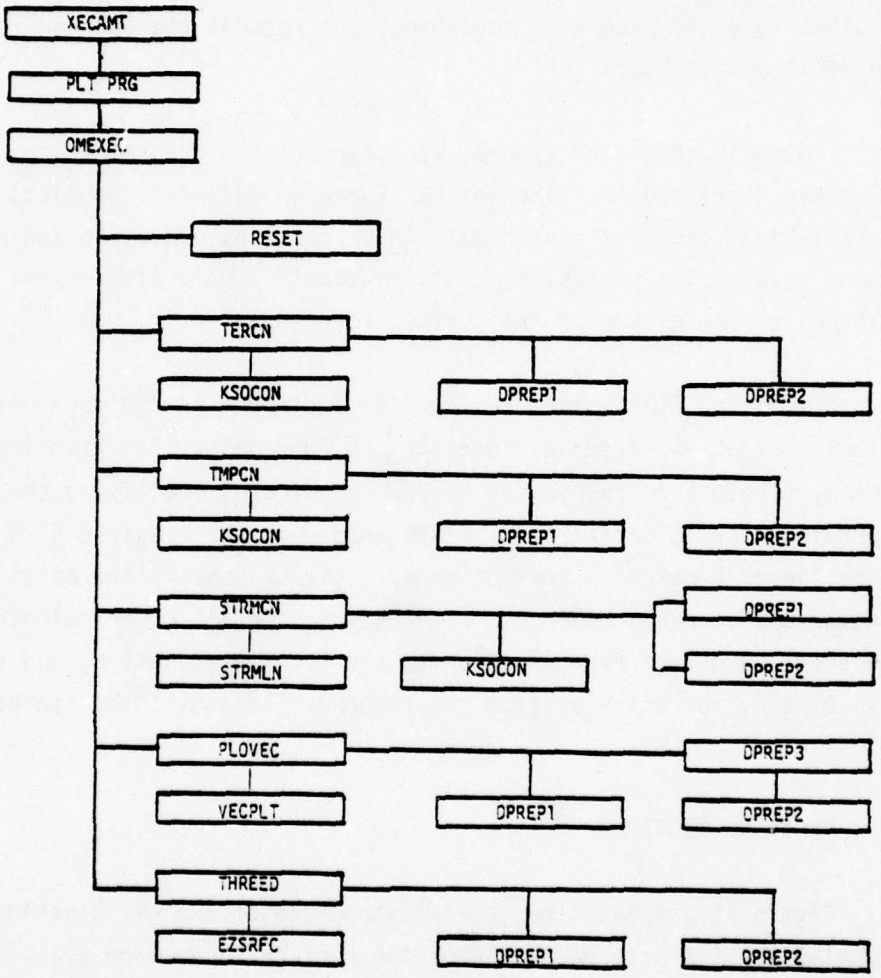


Figure 11. Organization of Plotting Program (PLTPRG)

structure.

The executive for the MRC plotting routine is OMEXEC which reads input, sets defaults, calls subordinates and prints formatted descriptions of the final resulting plots. RESET initializes the plotter prior to each new plot. Initialization may consist of overlaying plots, vertical stacking on the plot paper, or horizontal displacement on the plot paper to a new origin.

The plot routine can produce several different types of plots. Each of these plot types is generated by one of the major subordinate elements of OMEXEC. The major subordinate elements call many of the same routines. TERCN sets up a call to KSOCON to produce contour plots of terrain elevation. TMPCN performs a similar task to produce contour plots of potential temperature. STRMCN sets up data for a call to STRMLN which produces streamline plots and, optionally, calls KSOCON for an overlaid plot of wind speed isotachs. PLOVEC prepares data for a call to VECPLT which plots the windfield as small discrete vectors, whose direction is that of the wind and whose magnitude is proportional to wind speed. THREED calls EZSRFC to obtain three-dimensional projective views of terrain relief.

The routines DPREP1 and DPREP2 are used to transfer data from mass storage blocks to arrays required as calling arguments for the library plot routines KSOCON, STRMLN, VECPLT, and EZSRFC. DPREP3 performs scaling and alignment for PLOVEC.

VI.3 FILE MANAGEMENT

The routine BLKINI is a brief interactive routine used to initialize the header block of the microterrain files or the files of surface wind analysis results. When these files are first catalogued BLKINI must be used to write the header data which makes possible subsequent reads of the header. The first word of the header should be set equal to zero.

TERDOC and PLTDOC write out formatted descriptions of header data for the microterrain and surface wind files respectively.

VII GENERAL CONCLUSIONS

There remain some general aspects as well as some specific points relating to the surface wind analysis for EPAMS which warrant further discussion. Experience in the development of the model has indicated limitations and suggested possible avenues of improvement. These are examined below.

VII.1 Data Analysis Procedure

The high-resolution surface wind analysis was originally conceived as an extrapolative/interpolative method for a terrain-sensitive estimation of surface wind fields on the basis of a few real time measurements located in the simulation area. In the EPAMS system such measurements are presently unavailable, so initial input estimates are derived by analysis of meteorological observational data remote in space and sometimes remote in time. The data analysis procedure is a rational attempt to bridge a large gap in scale on a physical basis without the use of an intermediate (meso-) scale model. A simple mesoscale model incorporating gross regional topography might perform this function better.

The mesoscale model of DANARD (1977) with some modifications would be appropriate for this purpose. The Danard model accounts for topographic structure, surface diabatic effects and synoptic pressure gradients through a clever application of the pressure tendency equation, the equation of motion, and the thermodynamic energy equation in a relatively simple manner. One of its strengths is the ability to operate with only numerical prediction data or data from a single sounding. Data of this type, in fact in more detail, is generated from the EPAMS data base by the data analysis procedures for the surface layer wind calculation. Two possible weaknesses of the Danard model are its rough parameterization of the atmospheric boundary layer and its present structure as an initial value problem requiring integrations forward in time to achieve a balanced solution. The use of better boundary layer

parameterization as described by ARYA (1977) possibly in conjunction with the ideas of thermal layers as used in the ASL meso-scale layer model would go far to rectify the first of these problems. Casting of the model equations into steady state variational form instead of initial value problem form would not only remove same uncertainties with the time-dependent approach, but would also permit the natural incorporation of constraints arising from additional surface or sounding observations which might exist. Use of all observational data could only improve reliability. Data analysis through a mesoscale model could provide more reliable input to the high-resolution wind model as well as providing windfields at mesoscale resolution over a larger region.

VII.2 Surface Layer Analysis

In the development of the numerical algorithms for the surface layer analysis an effort was made to encode the basic model equations as accurately as possible. The analysis procedure is singularly delicate because the variational integral depends upon the fourth power of velocities and contains no terms such as opposing pressure and frictional forces whose balance would occur at a finite velocity. With some early algorithms the model run without any buoyancy forces produced wind fields which decreased in speed as the relaxation proceeded, with consequent monotonic decrease of the constraint to no minimum but zero. At first this behavior was attributed entirely to the "open boundary" problem of the surface layer approximation and a great many, necessarily somewhat simplified, remedies using an additional vertical layer were attempted. None of these efforts were satisfactory and in the end they proved unnecessary. The difficulty was overcome by the more accurate algorithm of averaging the square of acceleration rather than squaring the average of acceleration within the fundamental flux boxes of the computation (See Appendix B). In general, experience with the method indicates that accurate computational replication of basic principles is more effective than seemingly more expeditious computational short-cuts.

Incorporation of more vertical structure, however, might still be desirable, not to solve the above problem but to enhance the fidelity of results. As presently constructed the model shows terrain sheltering and wake structure but complete effects of flow separation are doubtful in an analysis of a single surface layer. Resolution above the present single layer could best be achieved by analytical functions with variable parameters which continue the surface layer profiles rather than by finite difference methods applied to several layers. Should further development of the model occur a complete re-examination of not only the vertical but also the horizontal structure of the computational grid would yield dividends. A uniform computational grid is wasteful in flat areas. A variable horizontal grid size, perhaps of basic triangular pattern, to provide resolution to only the degree needed would decrease overall storage requirements and free memory to incorporate more vertical structure.

The geometrical proportions of the basic flux box impose a limitation on the use of the surface layer analysis. The present analysis contains approximations valid if the layer thickness is less than 0.1 of the basic horizontal grid interval. The maintenance of this approximate proportion also insures that momentum fluxes through top and side faces of the flux box are roughly comparable. For validity of the atmospheric surface layer approximations the computational height must remain in the surface layer, perhaps less than 50 meters. Therefore uncertainties arise if a horizontal grid interval greater than 500 meters or much less than 10 times the computation height is used.

Because the basic wind analysis routine was developed on a smaller computer than the UNIVAC 1108, the numerical code performs more computations than necessary when more memory is available. At each grid point at each relaxational sweep physical quantities are re-computed at grid points of the surrounding 9-point template. The templates, of course, overlap and computation time could be saved by providing large global arrays to store these physical quantities and avoid redundant calculations. While faster running is desirable on the UNIVAC 1108, the adaptability of the basic wind analysis

(without the data accessing and processing routines) to smaller computers underscores its potential applicability for operational use in the field.

As mentioned previously treatment of the diabatic effects at the earth-atmosphere interface might be improved. Not only could the fluid dynamic energy equation be handled with more precision, but also better initial surface air temperature estimates are possible. Algorithms to account for solar radiative heating and long wave length ground radiation as a function of terrain slope, sun position and ground cover could be added to the present estimating procedure which is based solely upon terrain elevation. Cloud cover and humidity, presently not considered, would also bear upon this question.

Many of these suggested improvements could be possibly incorporated, but at the expense of increased model complexity. One of the prime advantages of the present surface layer analysis is its simplicity, both in concept and in its numerical implementation. Throughout the model development an effort has been made to refrain from non-productive elaboration, based on the understanding that simplicity was a desirable attribute. Refinements should be instituted only if clearly required and then only if they do not entail a heavy computational burden. The available information upon which the analysis is based may not justify attempts at added precision. Insightful approaches rather than brute-force computations are needed.

VII.3 Validation

Presently the surface layer wind analysis is unvalidated in the sense that its present form has not been compared with observational data. Since observational data at the resolution of the analysis will likely be extremely difficult to obtain any validation must consist of selected spot comparisons. The problem is the comparison of an estimated windfield, (at the model resolution) with another windfield well defined at observation locations but only estimated at other locations. A cross-correlation of the observed wind vectors

with the modeled wind vectors would provide a measure of comparison but only at the set of observational sites. The need is for a general integral measure to compare two fields taking account of the fact that except at selected points one of the fields, the observational one in this context, is the more poorly defined. We do no more than state this general conception of the validation problem.

However, the surface layer analysis within EPAMS presents at least three model structures to validate. The surface layer analysis itself given accurate driving parameters, the data analysis procedure for obtaining driving parameters from meteorological data types, and the total analysis which couples together both the data analysis and the surface layer analysis are the three structures. The overall analysis could be relatively invalid because of the inadequancy of one of its stages. Likewise the wind field analysis might operate well only if provided more accurate input than the data analysis generates. Investigation of these detailed questions is facilitated by the control structure provided, which allows separate operation of the data analysis routines and user supplied input to the basic surface layer analysis.

VII.4 Additional Model Uses

The surface layer analysis through its employment of surface layer wind and temperature profiles establishes in the course of the calculation the parameters of these profiles. Two parameters of interest are the surface friction velocity, u_*, and the Monin-Obukhov length, L. Presently these parameters are not saved in the analysis, but given adequate storage they could be recovered and stored in suitable arrays. These parameters are of interest in their own right and perhaps could provide necessary input for a calculation of flow within the vegetative canopy.

The advective acceleration and its accurate calculation play a fundamental role in the surface layer analysis. Once a state of minimum constraint

is reached the advective acceleration does not remain fixed but at each grid point continues to fluctuate about some average value as the relaxation continues. The fluctuating acceleration resulting from fluctuating velocities is dimensionally equivalent to a fluctuating stress gradient. To be sure this fluctuation is a fluctuation in relaxation steps rather than a fluctuation in time. Nevertheless the magnitude of this fluctuation is governed by the range of local dynamic imbalance occurring in the flow field. It is tempting to conjecture that these fluctuations may model the actual grid-resolvable turbulent fluctuations of the flow. If this possibility were explored and substantiated the surface layer analysis could be extended to supply turbulent diffusivities in flows over rough terrain in addition to the mean wind field itself.

The surface layer wind field analysis is a new approach to a difficult problem, an approach which offers significant conceptual and practical advantages. Further development should consist of extensive exercise of the model coupled with observational tests. Refinements and extensions as dictated by experimental and theoretical considerations can be introduced both to perfect the present model and to pursue additional avenues implicit in the concept.

VIII. BIBLIOGRAPHY

1. S.P.S. Arya, "Suggested Revisions to Certain Boundary Layer Parameterization Schemes Used in Atmospheric Circulation Models," Mon. Weather Rev. 105, 215 (1977).

2. P.E. Appell, "Les equations du movement d'un fluide parfait déduites de la consideration de l'énergie d'acceleration," Annali di Matematica pura ed applicata, Series 3, Vol. XX, 37-43 (1912).

3. J.A. Ball, "Physically-Based Estimation of Flow Fields from Limited Information, A High Resolution Model of Surface Wind in Broken Topography," Progress Report, Mission Research Corporation, Santa Barbara, CA, June 6, (1975).

4. A.K. Blackadar and H. Tennekes, "Asymptotic Similarity in Neutral, Barotropic, Atmospheric Boundary Layers," J. Atmos. Sci. 25, 1015-1020 (1968).

   H. Tennekes, "The Logarithmic Wind Profile," J. Atmos. Sci. 30, 234-238 (1974).

   _____, "Similarity Laws and Scale Relations in Planetary Boundary Layers," Chap. 5 of Workshop on Micrometeorology, D.A. Haugen (ed.), American Meteorological Society, Boston (1973).

5. P. Bradshaw, "The Analogy between Streamline Curvature and Buoyancy in Turbulent Shear Flow," J. Fluid Mech. 36, Part I, 177-191 (1969).

6. J.A. Businger, "Turbulent Transfer in the Atmospheric Surface Layer," Chap. 2 of Workshop on Micrometeorology, D.A. Haugen (ed.), American Meteorological Society, Boston (1973).

7. M. Danard, "A Simple Model for Mesoscale Effects of Topography on Surface Wind," Mon. Weather Rev. 105, 572 (1977).

8. R.K. Dumbauld and J.R. Bjorklund, "Mixing-Layer Analysis Routine and Transport/Diffusion Application Routine for EPAMS, "Research and Development Technical Report, ECOM-77-2, OSD-1366, U.S. Army Electronics Command, Fort Monmouth, N.J. (1977).

9. E. Guillaume, "Sur l'extension des équations mécaniques de M. Appell á la physique des milieux continus; application á la theorie des électrons," Comptes rendus 152, 875 (1913).

10. C. Lanczos, The Variational Principles of Mechanics, (2nd Edition), p. 106, University of Toronto Press, Toronto (1962).

11. J.L. Lumley and H.A. Panofsky, The Structure of Atmospheric Turbulence, p. 100, Wiley, New York (1964).

12. J. Serrin, "Mathematical Principles of Classical Fluid Mechanics," p. 144 in Handbuch der Physik, Vol. VIII/1, Springer-Verlag, Berlin (1959).

APPENDIX A

FILE DESCRIPTIONS AND USER INSTRUCTIONS

The total software associated with the surface layer wind analysis consists of the following:

1) A main program file, MRC*EPAMS. (hereafter referred to as M.)

2) A utility program file, MRC*UTILITY. (hereafter referred to as MT.)

3) Four data files:
   SWUS*MRCTRF
   SWUSMRC*WNDARCHIVE
   EURO*MRCTRF
   EUROMRC*WNDARCHIVE.

All software is stored on UNIVAC 1108 library tape. Program files contain the following types of elements:

1) FORTRAN source

2) Relocatable

3) Absolute

4) Job control

5) Sample input and documentation

File M. is used for the creation and execution of the surface wind

program, while MT. supplies plotting capability and terrain filtering routines. The first two data files contain, respectively, the micro terrain input and micro wind output for the southwest U.S. The other two data files contain corresponding data for Europe. The structure of the data files is as follows. The terrain files have a header in block 1, followed by pairs of blocks containing terrain and roughness data, respectively. The header consists of the 251 words in common block /TBLHDR/. The first word, RCOUNT, gives the number of archived pairs in the file. The other words give grid dimensions, UTM coordinates, and grid spacing for each pair of blocks. The surface wind and temperature output files also contain a header in block 1, followed by triples containing the output fields of potential temperature, u-component of wind, and v-component of wind. The headers for these files consist of the 401 words in common block /MRCHDR/. The first word gives the number of archived triples in the file. Other words give the driving parameters used for the run creating the output fields.

The surface wind model draws on the following data bases within the EPAMS system: Upper air, surface station, GWC prediction, mesomodel, and mesonet. As of November, 1977, these data sets were contained in the following files:

FOR SOUTHWEST U.S.

FILE NAME	DESCRIPTION
SWOBS.	upper air directory
SWUAD.	upper air data inventory
SWUS*SFDTA1.	surface station data inventory
SWGWCP.	GWC directory
SWGWCD.	GWC data inventory
MRC*TERRAIN.	mesoscale terrain
323*WSGRD2.	mesomodel output

FOR EUROPE

FILE NAME	DESCRIPTION
EUROBS.	upper air directory
EURUAD.	upper air data inventory
EURO*SFDTA1.	surface station data inventory

Comments: 1.) GWC and mesomodel data are not presently available for Europe. 2.) The surface station directory is contained in the first block of the surface station inventory rather than in a separate file. 3.) The connection between files and fortran logical unit variables is given below:

FILE	JCL USE NAME	FORTRAN LOGICAL UNIT NUMBER	FORTRAN VARIABLE
MRC*TERRAIN.	F3.	3	TRRNFL
323*WSGRD2.	F7.	7	GRIDF2
XINFIL.	10.	10	XINFIL
SWOBS.	F8.	8	OBFILE
SWUAD.	F1.	1	UAFILE
SWGWCP.	F11.	11	GWCOBF
SWGWCD.	F12.	12	GWCFIL
SWUS*SFDTA1.	F15.	15	SFCFIL
SWUS*MRCTRF.	F16.	16	MICTRF
SWUSMRC*WNDARCHIVE.	F17.	17	MOTFIL
PRINT$.		6	XOTFIL

4.) The input file, XINFIL., is a GUP file which may need to be created prior to running the model. 5.) The output file is PRINT$. which is the default

file for FORTRAN logical unit number 6. 6.) To run the model in Europe requires appropriate changes in file assignments. JCL elements which assign the proper files are M.WNDASG/SWUS for the SWUS and M.WNDASG/EUR for Europe.

MAIN PROGRAM -- INPUTS

The surface wind model program allows the user flexibility in the following areas:

1) data bases to be used,

2) time and place of simulation,

3) version of the wind model to the run.

The user specifies his options by setting flags in the input file (input data is in namelist format). The standard input file contains the namelists /XFLAGS/, /FLAGS/, and /TIMPLA/. A description of all flags in the input file is given in element M.USERGUIDE.

/XFLAGS/ sets flags held in common block /MESAGE/ which are used by the EPAMS executive. The flags are:

FUNCTN, TASK, JOB, BLOCK, FILE, PRINT, LEVEL.

The surface wind model may be run by setting FUNCTN = 2, TASK = 2, and JOB = Ø. Other flags are not used.

/FLAGS/ sets flags held in common block /MRCIPF/ and are used by WNDMGR to control the data base search and execution of the wind model. These flags are described below.

IDAT(I), IMDT(I), I = 1,6. These twelve flags specify the data bases to be searched and special features of the search. Indices one thru six correspond, respectively, to upper air, surface station, GWC, mesomodel, mesonet, and manual input data. If IDAT(I) = 0, a search is made of the data base corresponding to index I. If IDAT(I) is not zero, no search is made. For example, a typical run would have IDAT(1) = 0, IDAT(2) = 0, IDAT(3) = 0, IDAT(4) =1, IDAT(5) = 1, IDAT(6) = 1; this indicates that the upper air, surface station, and GWC data bases are to be searched for relevant data, while no use is to be made of mesomodel, mesonet, or manual input data. Mesonet data is not present at this time and calls to the mesonet accessing routine are in fact dummied out. Mesomodel data should be sought only if it has been specially created for the surface wind model. It is hoped that later versions of the EPAMS system would permit a mesomodel to be called prior to calling the surface wind program. If the user wishes to input manual data, no other data bases need be searched. The flags IMDT(I) give special information about how the data base searches are to be done. If IMDT(1) = 0, the UA inventory is searched for observations just preceding simulation time. If IMDT(1) = 1, UA observations just preceding and just following simulation time are sought and readings are interpolated in time. IMDT(1) should be zero for real-time simulations, and 1 for European simulations (since no GWC data exists in Europe). IMDT(2) places limits on the surface station data search. Data up to IMDT(2) hours old will be sought. The recommended value is IMDT(2) = 2. IMDT(4) is the number of the first block of mesomodel data in file 323*WSGRD2. Presently one should set IMDT(4) = 8 if mesomodel data is to be sought. Flags IMDT(3), IMDT(5), IMDT(6) are not used.

The flag ILASTR, if set equal to 1, directs WNDMGR to return control to the EPAMS executive at the end of a run. If ILASTR = Ø, WNDMGR will retain control and read input from namelists $FLAGS and $TIMPLA for another run.

IHIERC specifies the version of WINDEX to be executed. If IHIERC = Ø.

data bases are searched and data analysis is performed, but WINDEX is not called. IHIERC = 1 directs WINDEX to apply corrections to the initial wind-field but make no changes to the initial temeprature field. IHIERC = 2 directs changes to be made in both wind and temperature fields.

IFLUX specifies the number of flux boxes to be used computing the local contribution to the acceleration and temperature divergence residuals. IFLUX must be 2 or 4, and 2 is recommended.

IRELAX is the number of relaxation sweeps to be performed by WINDEX in searching for wind and temperature fields with minimum residual. IRELAX probably should be greater than the largest array dimension (i.e. larger than ILE + 1 - ILW and JLN + 1 - JLS). IRELAX = 50 is recommended.

ILE, ILW, JLN, JLS specify the portion of the micro terrain grid over which the wind and temperative fields are to be simulated. The terrain grid is blocked in as a matrix of dimension ILL by JLL (40-by-40 in SWUS, 51-by-31 in Europe). If one wishes to simulate the wind only in the left half of this matrix, one would set ILE = 20, ILW = 1, JLN = 40, JLS = 1. The default and recommended values are ILE = ILL, ILW = 1, JLN = JLL, JLS = 1.

The flag MTRBLK may be used to specify which set of micro terrain and roughness data (from file SWUS*MRCTRF. or EURO*MRCTRF.) is to be used. The micro terrain file contains a header in block 1, followed by pairs of blocks containing terrain and roughness data. For example, to use the fifth pair, set MTRBLK = 5. The alternative method is to set MTRBLK = Ø, in which case the header is searched for a block of terrain with southwest corner within 1Ø kilometers of location (XPLACE, YPLACE) (see below). If no such terrain is found, execution of WINDEX is suppressed.

NTST is used by routine DRIVRS to determine whether WINDEX should be executed. If NTST = 0, a search is made to see if the driving parameters of a previously archived run match the present parameters. If a match is found,

execution of WINDEX is suppressed. If NTST = 1, no search is made; WINDEX is executed and its output is archived.

/TIMPLA/ sets flags held in common block /COORD/. These flags are XPLACE, YPLACE, ZPLACE, YEAR, MONTH, DAY, HOUR, MINUTE. XPLACE, YPLACE are the UTM coordinate of the southwest corner of the simulation site. ZPLACE is the elevation in meters of the site. If ZPLACE = 0.0, the program will determine a default value. ZPLACE = 0.0 is recommended. XPLACE, YPLACE are used to locate the terrain block to be used. After the terrain block is found, XPLACE and YPLACE are redefined to agree with the actual coordinates of the southwest corner of the block found. YEAR, MONTH, DAY, and HOUR are integer variables which refer to the local time at which the simulation is to take place. MINUTE is not used.

It is possible to run the model independently of meteorological data bases by supplying the mesoscale driving parameters as input. The additional input is read from namelist /MFLAGS/, consisting of the following variables:

TPRO(1), TPRO(2), TPRO(3)	-	850,700,500 mb potential temperature profile
ZPRO(1), ZPRO(2), ZPRO(3)	-	850,700,500 mb height profile
DELQ	-	surface heating
GENWND(1)	-	easterly component of mesoscale wind
GENWND(2)	-	westerly component of mesoscale wind
NUMOBS	-	number of spot observations

If NUMOBS is nonzero, then the input namelist /MFLAGS/ must be followed by NUMOBS lines of data, formatted I5, I5, F10.2, F10.2, F10.2. The data from line I sets the following variables:

KINDX(I), LINDX(I) - grid indices of spot observation I

UW(I) - u component of wind

VW(I) - v component of wind

THETAØ(I) - potential temperature

Table A.1 summarizes information about input variables and defaults. Defaults are applied whenever an input value lies outside the range of the variable. If no range is specified, no default is given.

PLOTTING PROGRAM -- INPUTS

The utility plotting program may be used to produce plots of wind, temperature, and terrain fields. A description of input variables is given in M.USERGUIDE. Additional comments follow.

The plotting program is designed to plot data which has been archived in either the micro terrain file or the surface wind and temperature file. Flexibility in the plotting routine is achieved via input flags. The data to be plotted is specified by setting the flags WATFIL, ISTASH, and ITYPE. WATFIL should be 16 for terrain plots and 17 for wind and temperature plots. ISTASH specifies the pair or triple of blocks to be used (the program computes the proper block number depending on values of WATFIL, ISTASH, and ITYPE). ITYPE specifies the type of plot:

ITYPE = 1 contour plot of terrain

ITYPE = 2 contour plot of temperature

TABLE A.1

NAMELIST: /XFLAGS/

VARIABLE	TYPE	RANGE	DEFAULT	COMMENT
FUNCTN	INTEGER			= 2 for wind model
TASK	INTEGER			= 2 for wind model
JOB	INTEGER			= 0 for wind model
BLOCK	INTEGER			not used
FILE	INTEGER			not used
PRINT	INTEGER			not used
LEVEL	INTEGER			not used

NAMELIST: /FLAGS/

VARIABLE	TYPE	RANGE	DEFAULT	COMMENT
IDAT(1)	INTEGER	Ø-1	Ø	UA access
IDAT(2)	INTEGER	Ø-1	Ø	SS access
IDAT(3)	INTEGER	Ø-1	Ø	GWC access
IDAT(4)	INTEGER	Ø-1	1	mesomodel access
IDAT(5)	INTEGER	Ø-1	1	mesonet access
IDAT(6)	INTEGER	Ø-1	1	*manual input*
IMDT(1)	INTEGER	Ø-1	Ø	UA interpolation
IMDT(2)	INTEGER	Ø-3	2	Surface data time constraint
IMDT(3)	INTEGER			not used
IMDT(4)	INTEGER	1-15Ø	8	mesomodel block #
IMDT(5)	INTEGER			not used
IMDT(6)	INTEGER			not used
ILASTR	INTEGER	Ø-1	1	last run flag
NTST	INTEGER	Ø-1	Ø	archiving required flag
IFLUX	INTEGER	2 or 4	2	# of flux boxes
IHIERC	INTEGER	Ø-2	Ø	model complexity
MTRBLK	INTEGER	Ø-*	Ø	micro terrain run #
IRELAX	INTEGER	1-1ØØ	5	# of relaxation sweeps
ILE	INTEGER	1-ILL	ILL	grid index limit
ILW	INTEGER	1-ICC	1	grid index limit
JLN	INTEGER	1-JLL	JLL	grid index limit
JLS	INTEGER	1-JCC	1	grid index limit

TABLE A.1 (continued)

NAMELIST: /TIMPLA/

VARIABLE	TYPE	RANGE	DEFAULT	COMMENT
XPLACE	REAL			UTM X-coordinate (km)
YPLACE	REAL			UTM Y-coordinate (km)
ZPLACE	REAL	DEFAULTED IF INPUT AS Ø.Ø		elevation (m)
YEAR	INTEGER			should be 1974
MONTH	INTEGER	1-12		local month
DAY	INTEGER	1-31		local day
HOUR	INTEGER	0-24		local hour
MINUTE	INTEGER	0-6Ø		not used

NAMELIST: /MFLAGS/

VARIABLE	TYPE	RANGE	DEFAULT	COMMENT
TPRO(1)	REAL			85Ø MB potential temperature
TPRO(2)	REAL			7ØØ MB potential temperature
TPRO(3)	REAL			5ØØ MB potential temperature
ZPRO(1)	REAL			85Ø MB elevation
ZPRO(2)	REAL			700 MB elevation
ZPRO(3)	REAL			500 MB elevation
DELQ	REAL			surface heating
GENWND(1)	REAL			mesoscale u-wind
GENWND(2)	REAL			mesoscale v-wind

ITYPE = 3	streamlines of windfield with speed contours overlaid
ITYPE = 4	vector plot of windfield with scaled magnitudes
ITYPE = 5	projective view of terrain relief

The block size of the data is given by ILL, JLL; set ILL = 40, JLL = 40 for the S.W. U.S., and ILL = 51, JLL = 31, for Europe.

The values of ILE, ILW, JLN, JLS should be the same as those used in creating the wind and temperature output for plots of such data; for plots of terrain, they may be set to limit the region which is to be plotted (see the description of these variables in /FLAGS/).

NUMUP is used to stack plots vertically. If 12 inch paper is used on the plotter, NUMUP should be 1; for 24 inch paper, NUMUP may be 2 if PLH is not more than 9.0; for 36 inch paper, NUMUP may be 3 if PLH is not more than 9.0 and may be 2 if PLH is not more than 14.0.

PLH is the height in inches of the vertical axis.

SAMPLE is used for plotting a sampling of the data. If SAMPLE = 1, the full array is plotted. If SAMPLE = 2, a sampled array is plotted. SAMPLE = 2 is recommended for ITYPE = 4 plots if PLH is less than 15.0.

WATPEN determines the plotter pen color. Changing the color is desirable for overlaid plots. The code is 1 = black, 2 = red, 3 = blue. Recommended values are WATPEN = 1 *for contour plots*, = 3 for windfield streamlines or vector plots. Titles are plotted in red.

OVRLAY is the overlay flag. If OVRLAY = 0, the plotter is advanced to *a new region for the plot.* If OVRLAY = 1, the plot is laid over the previous

plot. ITYPE = 4 plots followed by ITYPE = 1 with OVRLAY = 1 plots are especially recommended.

NCON is the number of contours to be drawn for contour plots. The following values are recommended:

NCON = 12 for ITYPE = 1 or 2;

NCON = 0 or 4 for ITYPE = 3.
(NCON = 0 is permissible only for ITYPE = 3)

TEXT is a character array used for plotting titles. It is always program-determined at present.

ISTOP is a stopping flag whose use is optional. If ISTOP = 1, no plot is produced and the program terminates. If ISTOP = 0, input is read for another run; but if the end of the input file is reached, the program stops normally without error.

TERRAIN CREATING PROGRAMS -- INPUTS

The terrain creating program for the SWUS is designed to create 40-by-40 blocks of terrain and roughness data for output to file SWUS*MRCTRF. The input source of terrain data is file NI*13-10E. which contains 80-by-80 blocks of terrain with grid spacing approximately 63.5 meters (each block represents a region approximately 5.08 km square). The 80-by-80 blocks are aligned with the (1,1) point in the southwest corner, and with increasing values of the first coordinate corresponding to points farther to the north. File NI * 13 - 10E. contains over 400 such blocks. The terrain creating program reads inputs in namelist format from XINFIL. The namelist is /INPUT/, containing the variables UTMX, UTMY, NSQRX, NSQRY, IQUAD, and ISTOP. The program searches the header of file NI*13-10E. for a block with southwest corner within 14.5 km of UTMX, UTMY. If such a block B is found, then the region consisting of

NSQRX-by-NSQRY of the 80-by-80 blocks (with block B in the southwest corner) is filtered to produce the 40-by-40 output block. The output block is aligned with the (1,1) point in the southwest corner, and with increasing values of the first coordinate corresponding to points farther to the east. Portions of the 40-by-40 output block may contain irrelevant data; for instance, if NSQRX = 1 and NSQRY = 2, only the data is columns 1 through 20 is relevant. If NSQRX = 1, NSQRY = 1, and IQUAD is nonzero, a forty-by-forty quadrant will be chosen from the 80-by-80 input block B. The quadrant chosen is determined by the value of IQUAD:

IQUAD = 1	southwest quadrant
IQUAD = 2	southeast quadrant
IQUAD = 3	northeast quadrant
IQUAD = 4	northwest quadrant

The program reads input from /INPUT/, creates output blocks, and repeats, until the end of the input file is reached or a value ISTOP = 1 is read.

The program for creating terrain in Europe is much simpler. The existing data base is a single 101-by-51 block near Fulda, Germany with southwest corner at UTM coordinates (552.4 km, 5592.665 km), grid spacing 100 m., and oriented at an angle of + .89982 radians with respect to west-east. The program accepts as inputs from namelist /INPUT/ the variables UTMX, UTMY, and ISTOP. If possible, a 51-by31 block is output with southwest corner within 75 m of UTMX, UTMY, grid spacing 100 m, and aligned with the input block. If no such block fits within the input block, a message to that effect is printed. Input is read till the end of the data is found or a value of ISTOP = 1 is read.

EXECUTING ABSOLUTE ELEMENTS

Long runs of the surface wind model should be run batch. This may be easily done from a demand terminal by executing the following control statements:

```
@    USE M., MRC*EPAMS.

@    ASG, AZ M.

@    ADD M.BATCHINP
```

At this point, the user will be in the Ed processor with a listing of the input file XINFIL. displayed on the terminal. After editing the input file as desired, enter

```
@    ADD M.GOBATCH

@    START,/R   M. BATCH,, [RUNID], [PAN],, [TIMEST], [PAGEST]
```

this initiates the batch run.

A similar procedure (described in element M.USERGUIDE) may be used to execute the plot program in batch mode.

Creating and executing of all FORTRAN programs in files M. and MT. has been done with the aid of elements containing job control language statements or sample input data. JCL elements may be used to compile all routines used in a program, map relocatable elements into an absolute element, or assign files. Table A.2 summarizes the use of these special elements.

AD-A055 861 MISSION RESEARCH CORP SANTA BARBARA CALIF F/G 4/2
PHYSICALLY-BASED HIGH RESOLUTION SURFACE WIND AND TEMPERATURE A--ETC(U)
MAR 78 J A BALL, S A JOHNSON DAEA18-77-C-0043
UNCLASSIFIED MRC-R-7731-1-278 ERADCOM/ASL-CR-78-0043-1 NL

2 OF 3
ADA
055861

Table A.2

JCL AND SAMPLE INPUT ELEMENTS

ABSOLUTE ELEMENT	FORTRAN COMPILATION	MAP PROCESSOR	FILE ASSIGNMENT	SAMPLE INPUT
M.BIGKAHUNA/SWUS	M.WNDFOR/SWUS	M.WNDMAP/SWUS	M.WNDASG/SWUS	M.WNDINP/SWUS
M.BIGKAHUNA/EUR	M.WNDFOR/EUR	M.WNDMAP/EUR	M.WNDASG/EUR	M.WNDINP/EUR
MT.MRCPLT7	MT.PLTFOR	MT.PLTMAP7	MT.PLTASG	MT.PLTINP
MT.TERCRE	MT.TERFOR	MT.TERMAP	MT.TERASG	MT.TERINP
MT.TERCRE/EUR	MT.TERFOR/EUR	MT.TERMAP/EUR	MT.TERASG/EUR	MT.TERINP/EUR

APPENDIX B GRIDING AND CONSTRAINT CALCULATION ALGORITHMS

The computational volume of the surface layer analysis consists of the layer between the user selected computation height and the terrain surface. In order to evaluate the integrals of the model equations (II.11) the computational volume is divided by the griding scheme into local flux boxes (volume elements) as shown in Figure B.1. Each of these flux boxes has the same constant thickness, ξ_c the specified computation height, measured normal to the terrain surface. The surface layer wind components and the potential temperature, both at the computation height, are the fundamental field quantities carried by the calculation. We describe the algorithm for computing the contribution of an arbitrary flux box to each of the two integrals of the model equations. Since the typical surface normal dimension of a flux box ($\sim$ 10m) is an order of magnitude less than the typical lateral dimension ($\sim$ 100m), surface normal resolution is greater than surface parallel resolution, a feature augmented still more by exploiting known functional forms (the empirical profiles) for variations in the surface normal direction. Integrations in the surface normal direction can be performed analytically. Consequently the basic algorithm for a single flux box can be described as a finite element method in the surface normal direction and a finite difference method (in flux form) for the lateral directions. The volume elements are not rectangular, but warped to follow the terrain surface. The profile of their projection on the horizontal reference plane is, however, a square.

Figure B.1 also displays the basic grid scheme. Terrain elevations form a square grid oriented parallel to UTM grid north and east. Wind components and potential temperature are calculated on a staggered grid with points centered between adjacent terrain elevations. This arrangement permits accurate calculation of terrain slopes in each of two mutually perpendicular horizontal directions at every point of the wind grid. Internally the calculation uses the indicated wind components v^1 and v^2, rotated 45 degrees counter clockwise with respect to grid east and north. In this rotated coordinate system the basis vectors of the terrain following coordinate system of Section II.4.2 are defined at the locations of all wind vectors.

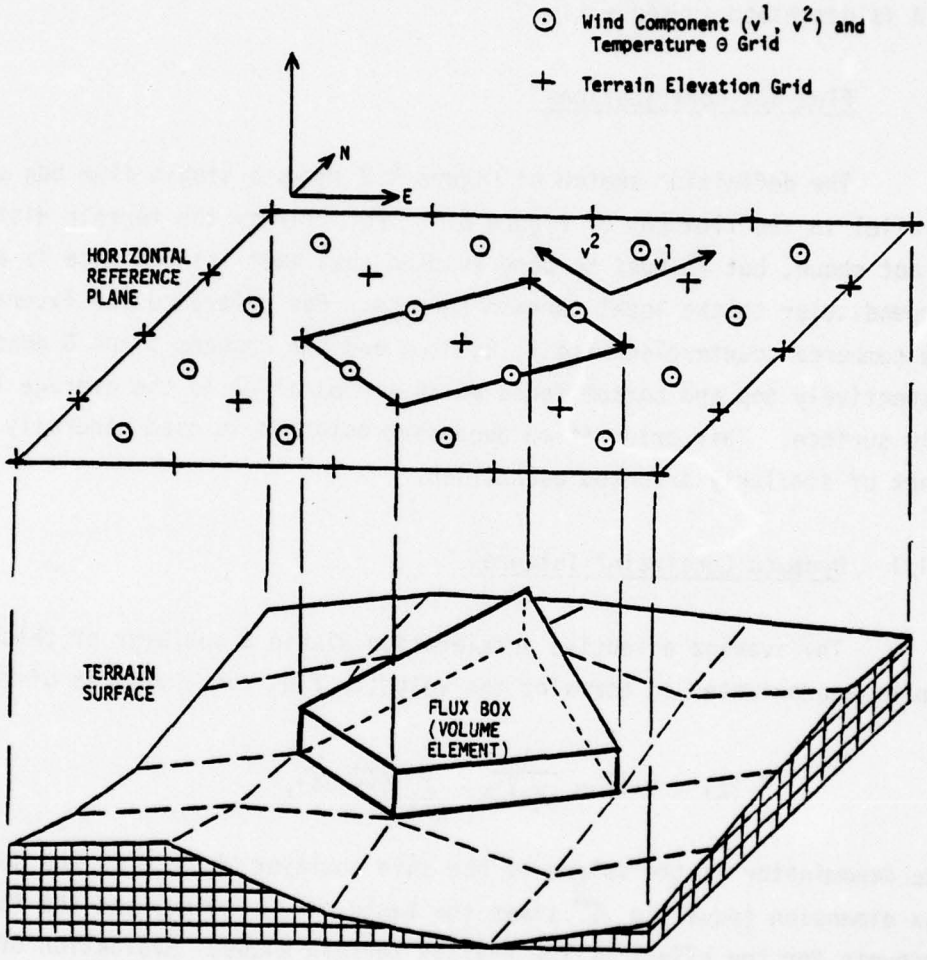


Figure B.1. Grid Structure and Element Geometry of WINDEX

In the following we derive the contribution of a single flux box to the constraint and to the temperature flux divergence integrals. The correspondence between local indexing and the global indexing of the simulation grid is described subsequently.

B.1 Flux Box Contributions

The definition sketch of Figure B.2 shows a single flux box oriented parallel to the flux box of Figure B.1. For clarity the terrain distortion is not shown, but it must be born in mind that each lateral face is a plane perpendicular to the local terrain surface. For reference the lateral faces are numbered counterclockwise 1, 2, 3, 4 and the numbers 5 and 6 denote respectively top and bottom faces which are parallel to the average local terrain surface. This orientation dependent notation is used generally to keep track of similarly oriented quantities.

B.1.1 Dynamic Constraint Integral

The average advective acceleration within a sublayer of thickness $\Delta\xi$ can be approximated in terms of the velocity flux out the faces of the layer.

$$\vec{A}(\xi) = \nabla\cdot\vec{\vec{vv}} \simeq \frac{1}{\Delta\xi d^2\sqrt{a}} \sum_{k=1}^{6} (\vec{\vec{vv}}\cdot\Delta\vec{s})_k$$

The denominator is the volume of the thin sublayer where d is the horizontal box dimension (equal to $\sqrt{2}$ times the basic grid spacing) and the factor $\sqrt{a}$ accounts for the effect of the average terrain slope. Evaluation of the terms in the sum requires use of the methods and notation of the warped terrain following coordinate system described in Section II.4.2. The index k denotes the faces of the thin sublayer.

The velocity flux dyadic $\vec{\vec{vv}}$ is the direct product of the velocity with itself. If $\vec{v} = v^1\hat{a}_1 + v^2\hat{a}_2 + v^3\hat{a}_3$,

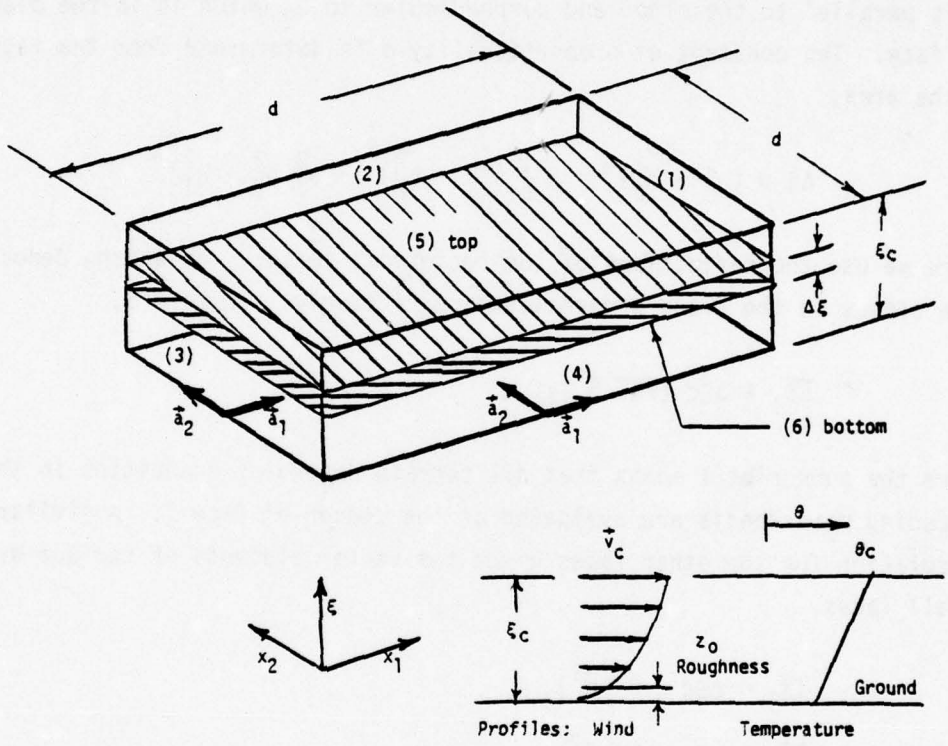


Figure B.2. Schematic Detail of Flux Box

$$\vec{v}\vec{v} = \sum_{i,j} v^i v^j \vec{a}_i \vec{a}_j. \qquad i,j = 1, 2, 3$$

The vector element of surface area $\vec{\Delta S}$ is in the outward normal direction of each face. On the top and bottom faces this is the direction of $\hat{a}_3$, and $-\hat{a}_3$, respectively. On a lateral face this direction is parallel to the surface and perpendicular to the surface parallel vector in the plane of the face. For example, consider face number 1. $\vec{\Delta S}_1 = \alpha \vec{a}^1$, since the reciprocal vector $\vec{a}^1$ is parallel to the slope and perpendicular to $\vec{a}_2$ which is in the plane of the face. The constant of proportionality α is determined from the magnitude of the area,

$$\Delta\xi\, d\, (1 + h_2^2)^{1/2} = \alpha \frac{1}{a} [(1 + h_2^2)^2 + h_1^2 h_2^2 + h_1^2]^{1/2}$$

where we use the definitions of the reciprocal vector. h_1 and h_2 denote terrain slopes in the 1 and 2 directions respectively. Therefore,

$$\vec{\Delta S}_1 = \Delta\xi d\, (\sqrt{a}\, \vec{a}^1)_1$$

where the subscript 1 means that all terrain dependent quantities in the preceding parenthesis are evaluated at the center of face 1. A similar calculation for the other faces gives the vector elements of surface area on all faces.

$$\vec{\Delta S}_1 = \Delta\xi d\, (\sqrt{a}\, \vec{a}^1)_1$$

$$\vec{\Delta S}_2 = \Delta\xi d\, (\sqrt{a}\, \vec{a}^2)_2$$

$$\vec{\Delta S}_3 = -\Delta\xi_d\, (\sqrt{a}\, \vec{a}^1)_3$$

$$\vec{\Delta S}_4 = -\Delta\xi d\, (\sqrt{a}\, \vec{a}^2)_4$$

$$\vec{\Delta S}_5 = d^2\, (\sqrt{a}\, \hat{a}_3)_5$$

$$\vec{\Delta S}_6 = -d^2\, (\sqrt{a}\, \hat{a}_3)_6$$

On faces 5 and 6 quantities must be evaluated using average values for the flux box.

The scalar products of the surface elements with the velocity flux summed over all faces of the sublayer give the result:

$$\begin{aligned}\vec{A}(\xi) = \frac{1}{d\sqrt{a}} \Bigg\{ & \left[v^1 v^1 \sqrt{a}\, \vec{a}_1 + v^2 v^1 \sqrt{a}\, \vec{a}_2 + v^3 v^1 \sqrt{a}\, \hat{a}_3 \right]_1 \\ & + \left[v^1 v^2 \sqrt{a}\, \vec{a}_1 + v^2 v^2 \sqrt{a}\, \vec{a}_2 + v^3 v^2 \sqrt{a}\, \hat{a}_3 \right]_2 \\ & - \left[v^1 v^1 \sqrt{a}\, \vec{a}_1 + v^2 v^1 \sqrt{a}\, \vec{a}_2 + v^3 v^1 \sqrt{a}\, \hat{a}_3 \right]_3 \\ & - \left[v^1 v^2 \sqrt{a}\, \vec{a}_1 + v^2 v^2 \sqrt{a}\, \vec{a}_2 + v^3 v^2 \sqrt{a}\, \hat{a}_3 \right]_4 \\ & + \frac{d}{\Delta\xi} \left[v^1 v^3 \sqrt{a}\, \vec{a}_1 + v^2 v^3 \sqrt{a}\, \vec{a}_2 + v^3 v^3 \sqrt{a}\, \hat{a}_3 \right]_5 \\ & - \frac{d}{\Delta\xi} \left[v^1 v^3 \sqrt{a}\, \vec{a}_1 + v^2 v^3 \sqrt{a}\, \vec{a}_2 + v^3 v^3 \sqrt{a}\, \hat{a}_3 \right]_6 \Bigg\} \end{aligned} \tag{B.1}$$

The condition of mass conservation, $\sum_k (\vec{v} \cdot \vec{\Delta S})_k = 0$, similarly approximated in flux form for the sublayer, gives the result,

$$(v^3 \sqrt{a})_5 - (v^3 \sqrt{a})_6 = \frac{-\Delta\xi}{d} \left[(v^1 \sqrt{a})_1 + (v^2 \sqrt{a})_2 - (v^1 \sqrt{a})_3 - (v^2 \sqrt{a})_4 \right] = \frac{\Delta\xi}{d} w$$

where the abbreviated notation w denotes the negative of the quantity in square brackets. Since slope dependent quantities are the same for faces 5 and 6 (invariance of the coordinate system in the normal direction is assumed) and slope parallel velocity components differ little across the thin sublayer we

use the divergence condition to combine the last two terms of Equation (B.1). On faces 1 through 4 terms of Equation (B.1) in v^3 are of order $(\xi_c/d) < 0.1$ with respect to other terms, a fact also deducible from the divergence condition. Dropping terms of this order throughout, the average acceleration at a height ξ in the flux box depends only upon the surface parallel components of velocity at that height. A further step is now to express the basis vectors $\vec{a}_1$ and $\vec{a}_2$ in terms of Cartesian components so terms may be combined.

$$\vec{A}(\xi) = \frac{\hat{i}}{d\sqrt{a}} \left\{(v^1v^1\sqrt{a})_1 + (v^1v^2\sqrt{a})_2 - (v^1v^1\sqrt{a})_3 - (v^1v^2\sqrt{a})_4 + (v^1w\sqrt{a})_5\right\}$$

$$+ \frac{\hat{j}}{d\sqrt{a}} \left\{(v^2v^1\sqrt{a})_1 + (v^2v^2\sqrt{a})_2 - (v^2v^1\sqrt{a})_3 - (v^2v^2\sqrt{a})_4 + (v^2w\sqrt{a})_5\right\}$$

$$+ \frac{\hat{k}}{d\sqrt{a}} \left\{[\sqrt{a}\,(v^1v^1h_1 + v^2v^1h_2)]_1 + [\sqrt{a}\,(v^1v^2h_1 + v^2v^2h_2)]_2\right.$$

$$- [\sqrt{a}\,(v^1v^1h_1 + v^2v^1h_2)]_3 - [\sqrt{a}\,(v^1v^2h_1 + v^2v^2h_2)]_4$$

$$\left. + [\sqrt{a}\,w\,(v^1h_1 + v^2h_2)]_5\right\} \qquad (B.2)$$

The quantity of interest is the square of the acceleration integrated over the volume of the flux box. Squaring the above produces many terms, all of which are of fourth power in the velocity components. If all velocity components obey the power law profile, Equation (II.7), a typical term in the integral involves the surface normal integral of the fourth power of velocity components,

$$\int_0^{\xi_c} (v)^4 d\xi = \frac{\xi_c}{4n+1}\, v_c^4$$

where v_c denotes the velocity component at the computational level which is carried by the calculation. Consequently the appropriate profile averaging of root mean square acceleration in the flux box is achieved by replacing all velocities in Equation (B.2) by their values at the computational level and by dividing all terms quadratic in velocity components by $\sqrt{4n+1}$ where n is the exponent of the local power law profile.

$$\vec{A}_{rms} = \frac{\hat{i}}{d\sqrt{a}} \left\{ (v_c^1 \, v_c^1 \, \sqrt{a} \; / \; \sqrt{4n+1})_1 + (\quad)_2 + \cdots \right\}$$
$$+ \frac{\hat{j}}{d\sqrt{a}} \left\{ (v_c^2 \, v_c^1 \, \sqrt{a} \; / \; \sqrt{4n+1})_1 + (\quad)_2 + \cdots \right\}$$
$$+ \frac{\hat{k}}{d\sqrt{a}} \left\{ [\sqrt{a} \; (v_c^1 \, v_c^1 \, h_1 + v^2 v^1 h_2) \; / \; \sqrt{4n+1}\,]_1 + \cdots \right.$$
$$\left. + [\sqrt{a} \; w_c \, (v_c^1 \, h_1 + v^2 v^1 h_2) \; / \; \sqrt{4n+1}\,]_5 \right\} \qquad \text{(B.3)}$$

The square of this quantity multiplied by the flux box volume provides the estimate of the acceleration squared integrated over the flux box volume which takes into account the surface normal resolution provided by the power law profiles. Averages of values of both slopes and wind components at faces 1, 2, 3, 4 are used to evaluate the term with subscript 5.

A less elaborate method of averaging is used for the buoyancy term. The average buoyance in the flux box is an average over the four lateral faces of the average buoyancy on each face. The average buoyance on each face is determined from the average temperature, where the average temperature $\langle\Theta\rangle$ on a face is,

$$\langle\Theta\rangle = \Theta_c - \tfrac{1}{2} \frac{d\Theta}{d\xi}\Big)_c \xi_c$$

a formula which uses the linear temperature gradient, Equation (II.10). The average buoyancy acceleration on each face of the flux box,

$$\vec{B} = \frac{\langle\Theta\rangle - \Theta_o}{\Theta_o} \; g \, \hat{k}$$

is a vertical vector. The average $\langle\vec{B}\rangle$ over the four lateral faces is used as the buoyancy acceleration throughout the flux box.

A fully three dimensional form of the contribution of each flux box to the constraint integral could be approximated as,

$$d^2 \xi_c \; \sqrt{a} \; (\vec{A}_{rms} - \langle\vec{B}\rangle)^2$$

where the leading factors are the flux box volume. However according to Section II.4.1 and Equation (II.11) only surface parallel buoyancy is used. This is obtained by subtracting out the surface normal component,

$$\langle \vec{B} \rangle_{11} = \langle \vec{B} \rangle - \hat{a}_3 \, (\langle \vec{B} \rangle \cdot \hat{a}_3)$$

a manipulation easily performed by expressing $\hat{a}_3$ in Cartesian components.

Consequently the numerical approximation for the local contribution of a flux box to the dynamic constraint integral, Equation (II.11), is:

$$d^2 \, \xi_c \, \sqrt{a} \, \left[\vec{A}_{rms} - \langle \vec{B} \rangle_{11} \right]^2 . \qquad (B.4)$$

These formulas are evaluated in the program segment ACCSQL which returns a quantity proportional to Equation (B.4), as well as the normal component of acceleration. To avoid repeated multiplication by constant factors ACCSQL includes in the acceleration calculation only the factors enclosed by curly brackets in Equation (B.3), with appropriate modifications of the buoyancy term.

B.1.2 Temperature Flux Integral

Since the surface normal resolution of temperature is coarser than the resolution of wind shear, and since model use of temperature is less precise than the treatment of dynamic quantities, the computation of temperature flux divergence does not divide a flux box into sublayers. A number proportional to the local contribution to the advective consistency integral, Equation (II.11), is approximated as,

$$\int_{Box} \left[\nabla \cdot (\vec{v}\Theta) \right]^2 dV \sim \frac{1}{\sqrt{a}} \left[\sum_{k=1}^{5} (\vec{v}\Theta \cdot \Delta\vec{S})_k \right]^2$$

where the index k runs over the all box faces except the bottom ($\vec{v}$ = o there) and each term in the sum is evaluated for average values on the face.

Using the formulas and notation derived in the preceding section to perform the scalar multiplication the sum becomes,

$$S_T = \sum_{k=1}^{5} (\vec{v}\Theta \cdot \overrightarrow{\Delta S})_k = d\xi_c \left[(\sqrt{a}\ \underline{v^1\Theta})_1 + (\sqrt{a}\ \underline{v^2 0})_2 - (\sqrt{a}\ \underline{v^1\Theta})_3 - (\sqrt{a}\ \underline{v^2\Theta})_4 \right] + d^2\ (\sqrt{a}\ v^3 0)_5$$

where the underlined quantities denote averages over a box face. The average surface normal velocity on the top face of the box is given by the continuity condition.

$$(\sqrt{a}\ v^3)_5 = -\frac{\xi_c}{d} \left[(\sqrt{a}\ \underline{v^1})_1 + (\sqrt{a}\ \underline{v^2})_2 - (\sqrt{a}\ \underline{v^1})_3 - (\sqrt{a}\ \underline{v^2})_4 \right]$$

If we denote the average temperature on the top face by $\langle\Theta_c\rangle$, the average of computational level temperatures at the four sides of the box, the temperature divergence sum requires only four terms.

$$S_T = \xi_c d \left\{ \left[\sqrt{a}\ (\underline{v^1\Theta} - \underline{v^1}\ \langle\Theta_c\rangle) \right]_1 + \left[\cdots \right]_2 \left[\cdots \right]_3 \left[\cdots \right]_4 \right\}$$

The averages over the lateral box faces are computed in terms of computational level quantities using the profiles of wind velocity and temperature, Equation (II.7). The power law wind profile gives for any wind component,

$$\underline{v^i} = V_c^i/(n+1).$$

Surface normal integration of the product of the wind profile and the temperature profile furnishes the formula,

$$\underline{v^i\Theta} = v_c^i\ \Theta_c/(n+1) - v_c^i\ \xi_c \left.\frac{d\Theta}{d\xi}\right)_c /(n+1)\ (n+2),$$

for the average components of temperature flux on the lateral faces.

The program segment TPFLXL employs these formulas to calculate the sum of the four terms, omitting constant multiplicative factors. This sum squared is returned as a measure of the square of the temperature flux divergence in the local flux box.

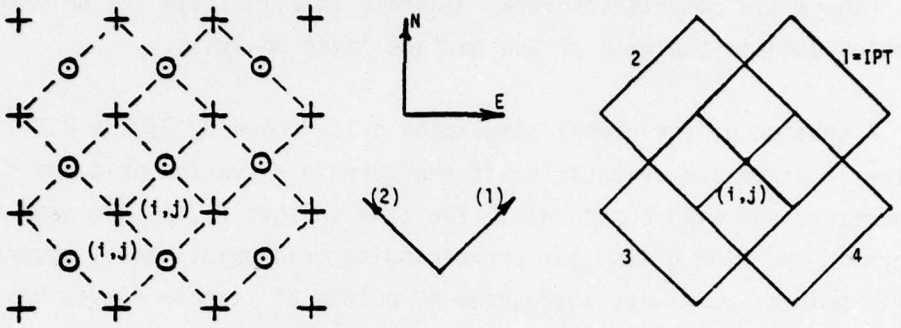
B.2 Griding and Indexing

Figure B.3 comprises several diagrams to illustrate the heirarchy of indexing used in the grid of the surface layer analysis.

A section of the global simulation grid, shown by Figure B.3.a, displays the location and orientation of the terrain elevation grid and the grid of temperature and wind components. The same indices (i, j) are used for both terrain and wind grids, but corresponding grid points are staggered. Wind grid points are always surrounded by points of terrain elevations, so the terrain grid contains one more point in each direction than the wind grid. Choosing the basic coordinate directions (1) and (2) as shown enables terrain slopes, h_1 and h_2, in these directions to be most accurately determined at each point of the wind grid. Each point of the wind grid (i, j) appears on a face of each of 4 possible flux boxes, which are also indicated.

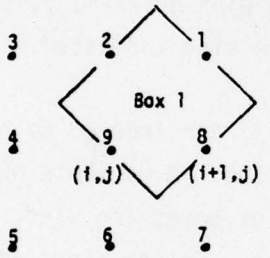
The flux boxes surrounding a point (i, j) are indexed by numbers 1, 2, 3, 4 with the orientation shown in Figure B.3.b. A variable NSET(•), indexed by these box indices, determines which flux boxes are used. If NSET = 1 the box is used, if NSET = 0 the box is not used. On the sides and corners of the grid NSET must equal zero for some boxes which are outside the grid. This decision is made by the program element STGDBX. At interior points of the grid all four boxes may be used or only the two boxes most nearly upwind of the point (i, j). Use of only upwind boxes is recommended to replicate advective phenomena. This decision is made in the program element WEIGHT according to the user set flag IFLUX. The box index, denoted IPT in the routines ACCSQL and TPFLXL, determines which of the four local flux boxes are used.

The four local flux boxes together embrace a total of 9 computational points, including the point (i, j). A local template indexing these points 1 through 9 is established as shown in Figure B.3.c. When the relaxation calculation reaches the grid point (i, j), required physical quantities are filled



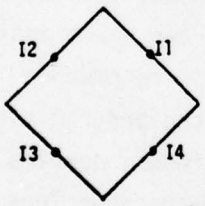
(a) Global Grid
+Terrain Elev., ⊙ Wind, Temp.

(b) Local Box Indices, IPT



Global Index	Template Index
i+1,j+1	1
i,j+1	2
i-1,j+1	3
i-1,j	4
i-1,j-1	5
i,j-1	6
i+1,j-1	7
i+1,j	8
i,j	9

(c) Local Template Indices



	Box Face			
	1	2	3	4
Box Index	I1	I2	I3	I4
1	1	2	9	8
2	2	3	4	9
3	9	4	5	6
4	8	9	6	7

(d) Template Indices of Box Faces

Figure B.3. Local Griding of WINDEX

into 9-dimensional arrays stored in the common block LOCAL. The correspondence between global indices (i, j) and the local template indices is established by the program segment LOCQUA as shown by the table of Figure B.3.c.

The analysis for a single flux box described in the preceding section uses a subscript notation 1 through 4 to denote the lateral faces of the flux box. Internally in the routines ACCSQL and TPFLXL these indices are denoted by the mnemonic symbols I1, I2, I3, and I4. These symbols are actual indices having values 1 through 9 to designate positions on the local template. The box index in conjunction with the box face designation determines the template index required to access physical variables on each face. This correspondence is shown by the table of Figure B.3.d, which is implemented by means of a data statement in the routines ACCSQL and TPFLXL.

APPENDIX C

ILLUSTRATIVE ANALYSIS RESULTS

This appendix contains a series of computer generated plots illustrating the output of the surface wind analysis performed by EPAMS for nine (9) cases: three (3) for a 3 km by 5 km area near Fulda, Germany, and six (6) for 10 km by 10 km areas (Soledad and Bear Canyons) on the White Sands Missile Range and at El Paso, Texas. In all cases, the input parameters of the surface layer analysis were obtained by automatic processing of meteorological data stored in the EPAMS data base, either the European data base or the SW United States data base, as appropriate.

Each figure presents one analysis and consists of two plots and a caption. The first plot, designated (a), shows scaled wind vectors overlaid on terrain contours, with contour elevations in meters. The second plot, designated (b), shows streamlines of wind direction and isotachs of wind speed in meters/second. On this plot, streamlines are drawn to obtain approximate uniform coverage, so their density has no relation to the wind speed. Stagnation points are designated by a small box. The caption of each figure lists the input parameters of that analysis.

The area in Germany is rolling and broken with forested hill tops and cultivated valleys. The maximum difference in elevation within the area is 130 meters. In this area, the wind analysis uses a horizontal grid interval of 100 meters and a computation height at which wind is estimated of 6 meters. It should be noted that Figures C-1, C-2, and C-3 which pertain to this area are not aligned with geographic north and east.

The El Paso and White Sands Missile Range areas are desert mountains and canyons with a maximum elevation difference within these areas as large as 1300 meters. The wind analyses for these areas use a horizontal grid interval of 254 meters and a computation height of 6 meters. The six cases for these areas are presented in Figures C-4, C-5, C-6, C-7, C-8, and C-9.

FIGURE C-1

Location: near Fulda, Germany.

Time: 1400 local standard time, 1 November 1974

Input Data		Source
Surface Heating ($\Delta Q/c_p$)	+4.255 K	Surface stations
Wind Direction	240 degrees	Closest surface station
Wind Speed	3.95 m/sec.	Geostrophic analysis of surface data

Profiles:

Pressure (mb)	Height (m,MSL)	Pot. Temp (K)	
850	1425	280.8	Spatial interpolation of 4 upper air soundings taken one hour previous to time of analysis.
700	2926	287.4	
500	5428	301.3	

A moderate breeze with heating of surface air in afternoon conditions.

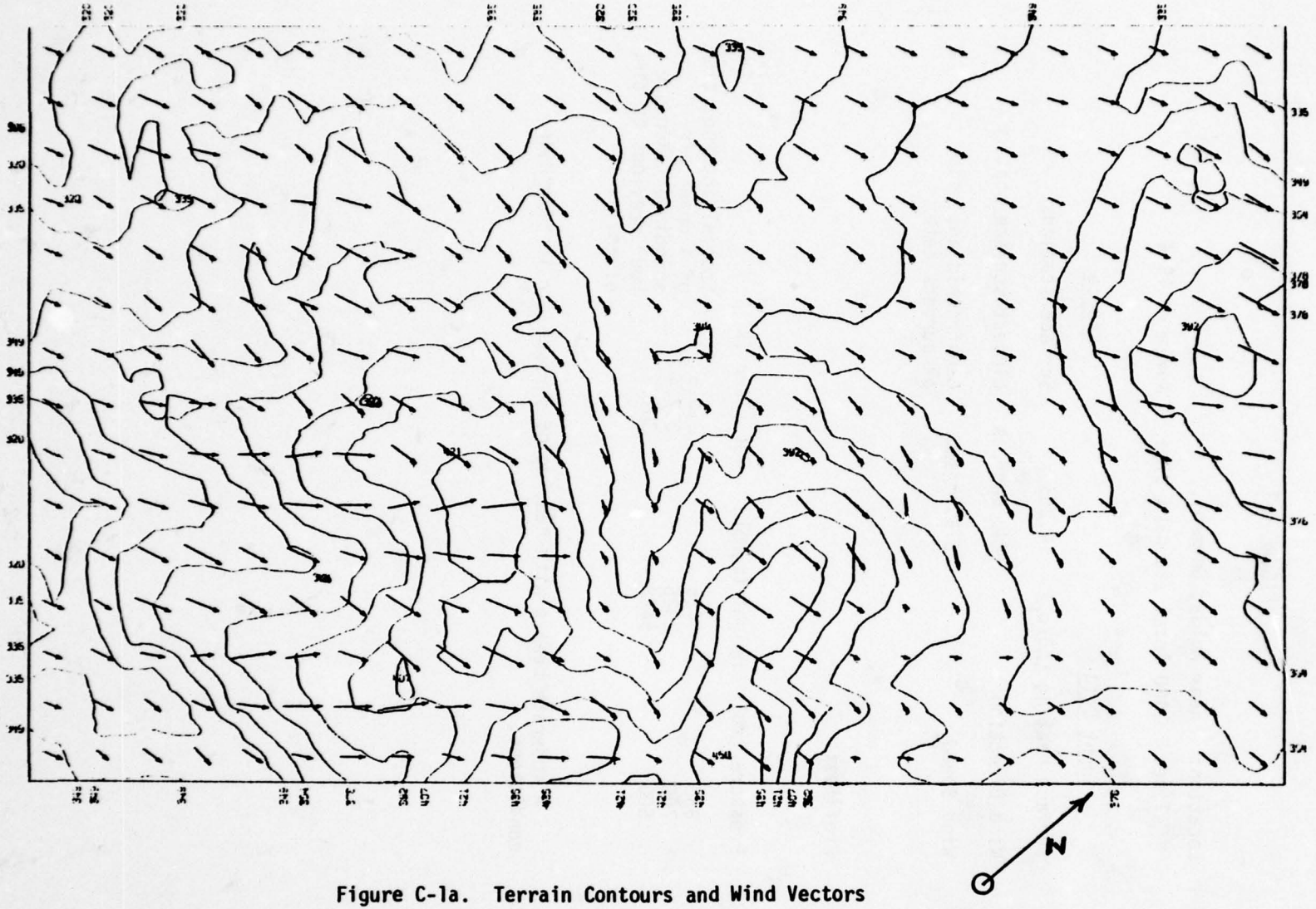


Figure C-1a. Terrain Contours and Wind Vectors

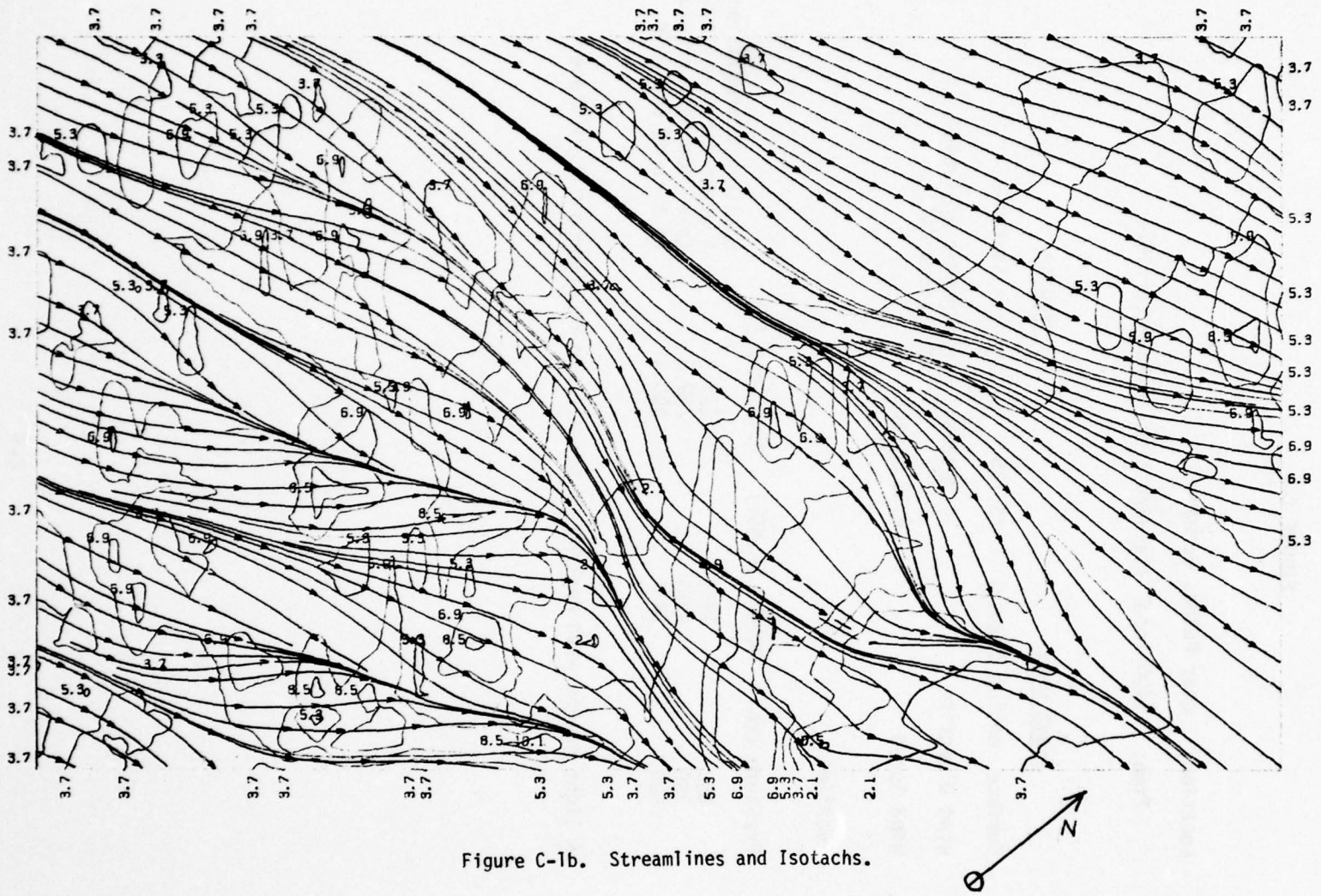


Figure C-1b. Streamlines and Isotachs.

FIGURE C-2

Location: near Fulda, Germany.

Time: 1400 local standard time, 4 November 1974

Input Data		Source
Surface Heating ($\Delta Q/c_p$)	+3.725	Surface Stations
Wind Direction	74 Degrees	Geostrophic analysis of surface data
Wind Speed	2.32 m/sec.	

Profiles:

Pressure (mb)	Height (m,MSL)	Pot. Temp (K)
850	1447	283.3
700	2961	289.3
500	5475	300.8

Spatial interpolation of 4 upper air soundings taken one hour previous to time of analysis.

A light breeze with heating of surface air in afternoon conditions.

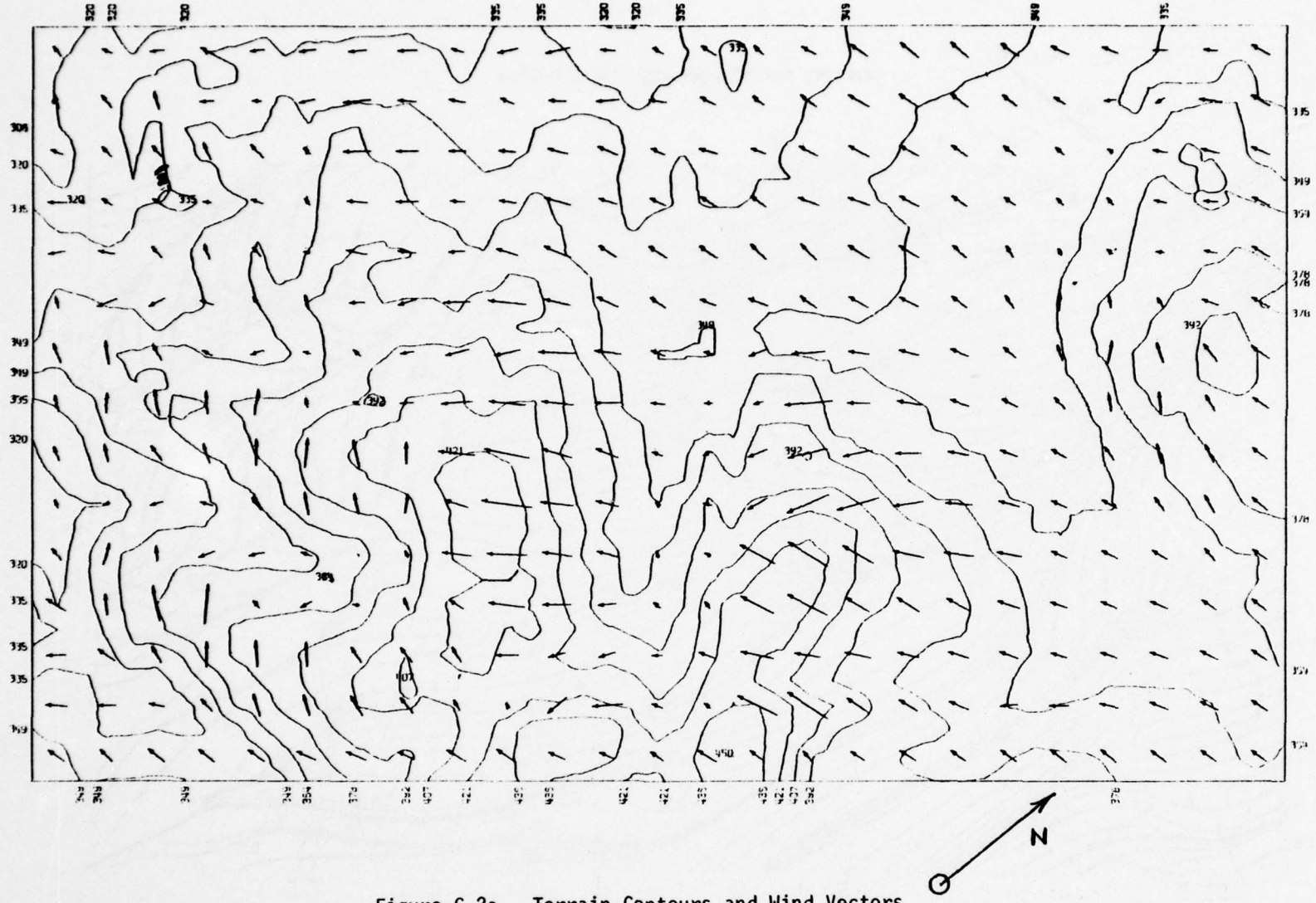


Figure C-2a. Terrain Contours and Wind Vectors

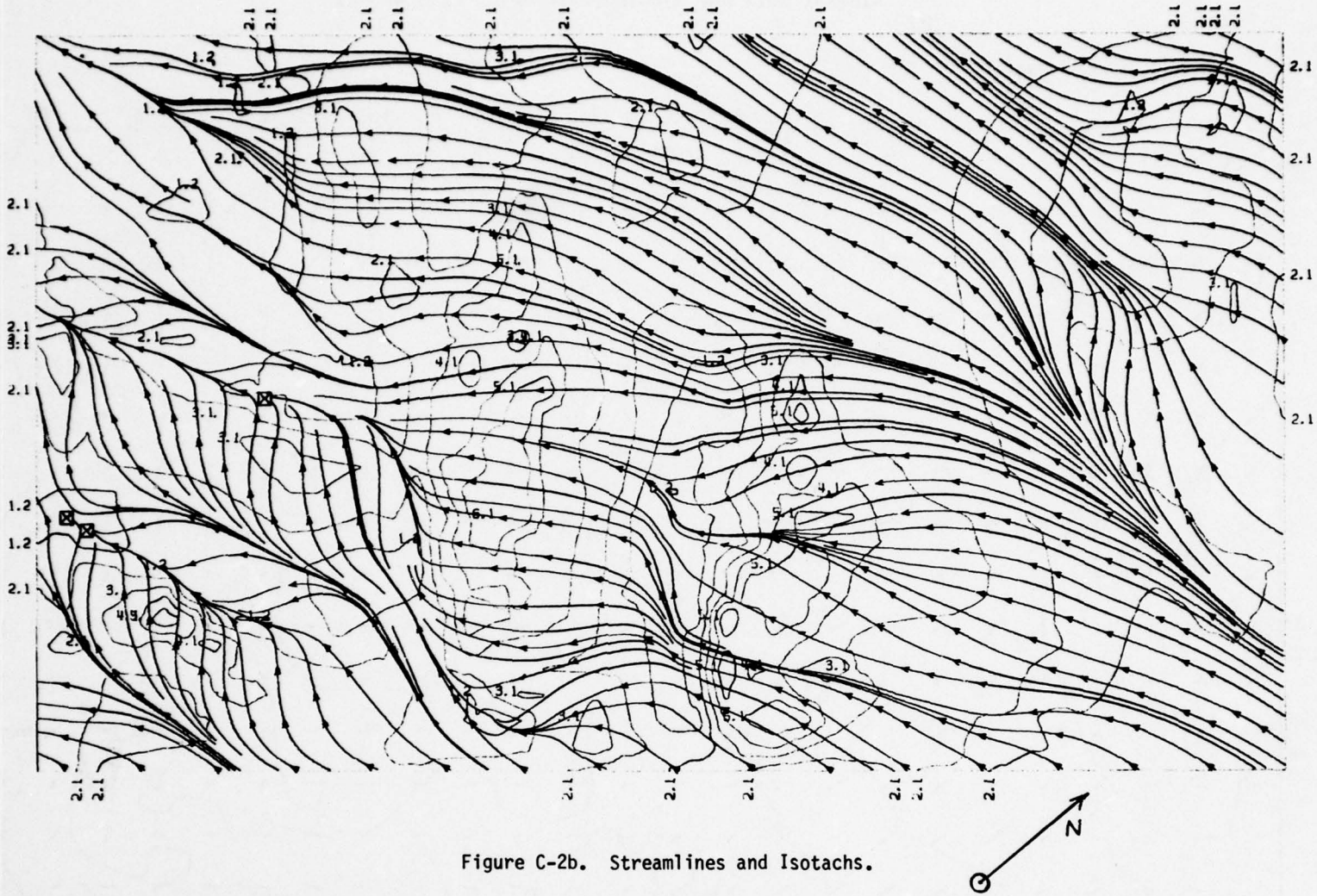


Figure C-2b. Streamlines and Isotachs.

FIGURE C-3

Location: near Fulda, Germany

Time: 0700 local standard time, 8 November 1974

Input Data		Source
Surface Heating ($\Delta Q/c_p$)	-6.33 k	Surface stations
Wind Direction	140 degrees	Closest surface station
Wind Speed	2.00 m/sec.	Closest surface station

Profiles:

Pressure (mb)	Height (m,MSL)	Pot. Temp (K)	
850	1553	286.2	Spatial interpolation of 4 upper air soundings taken at time of analysis
700	3087	293.9	
500	5638	306.3	

Drainage flow of negatively buoyant surface air under light winds in the early morning.

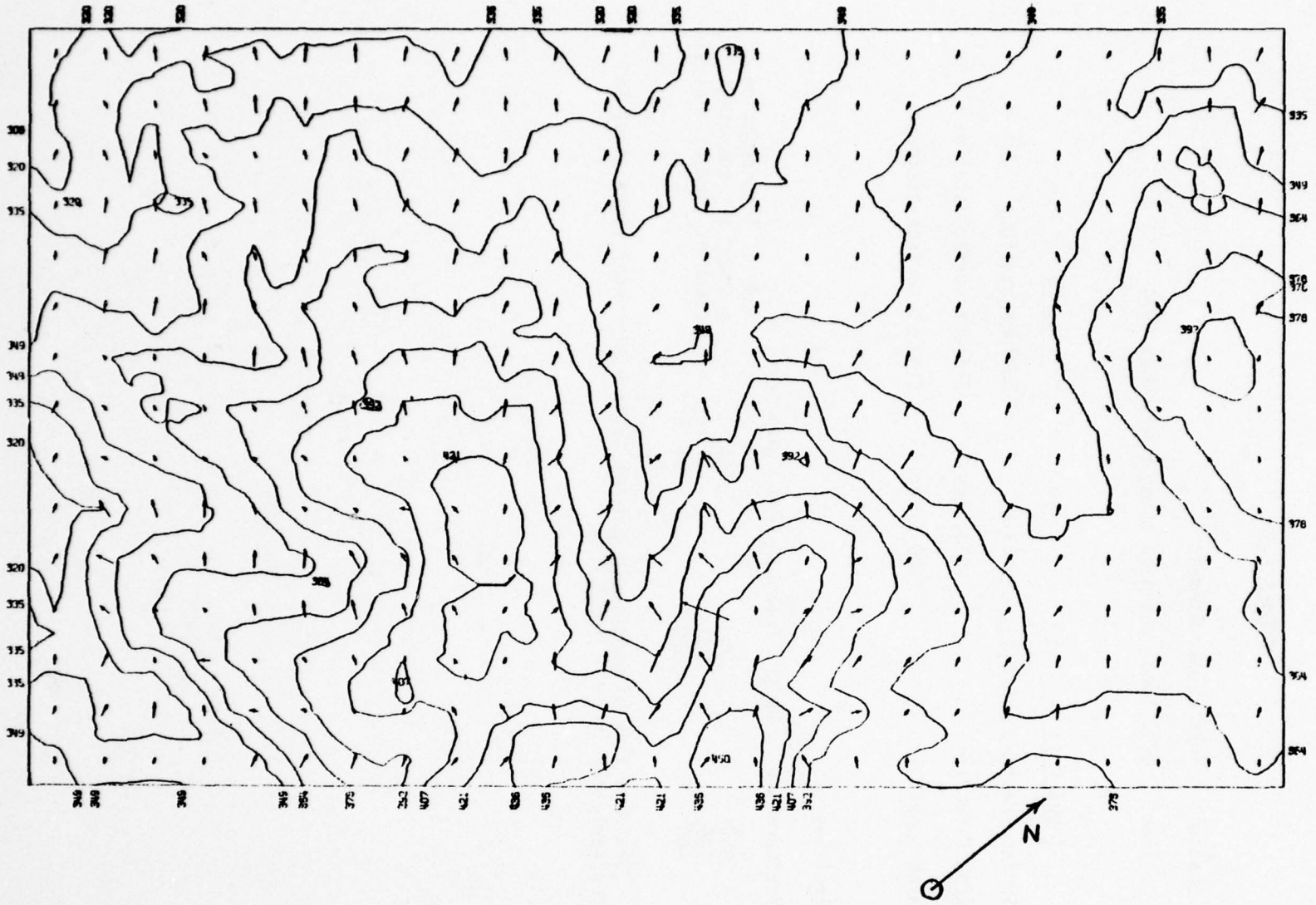


Figure C-3a. Terrain Contours and Wind Vectors

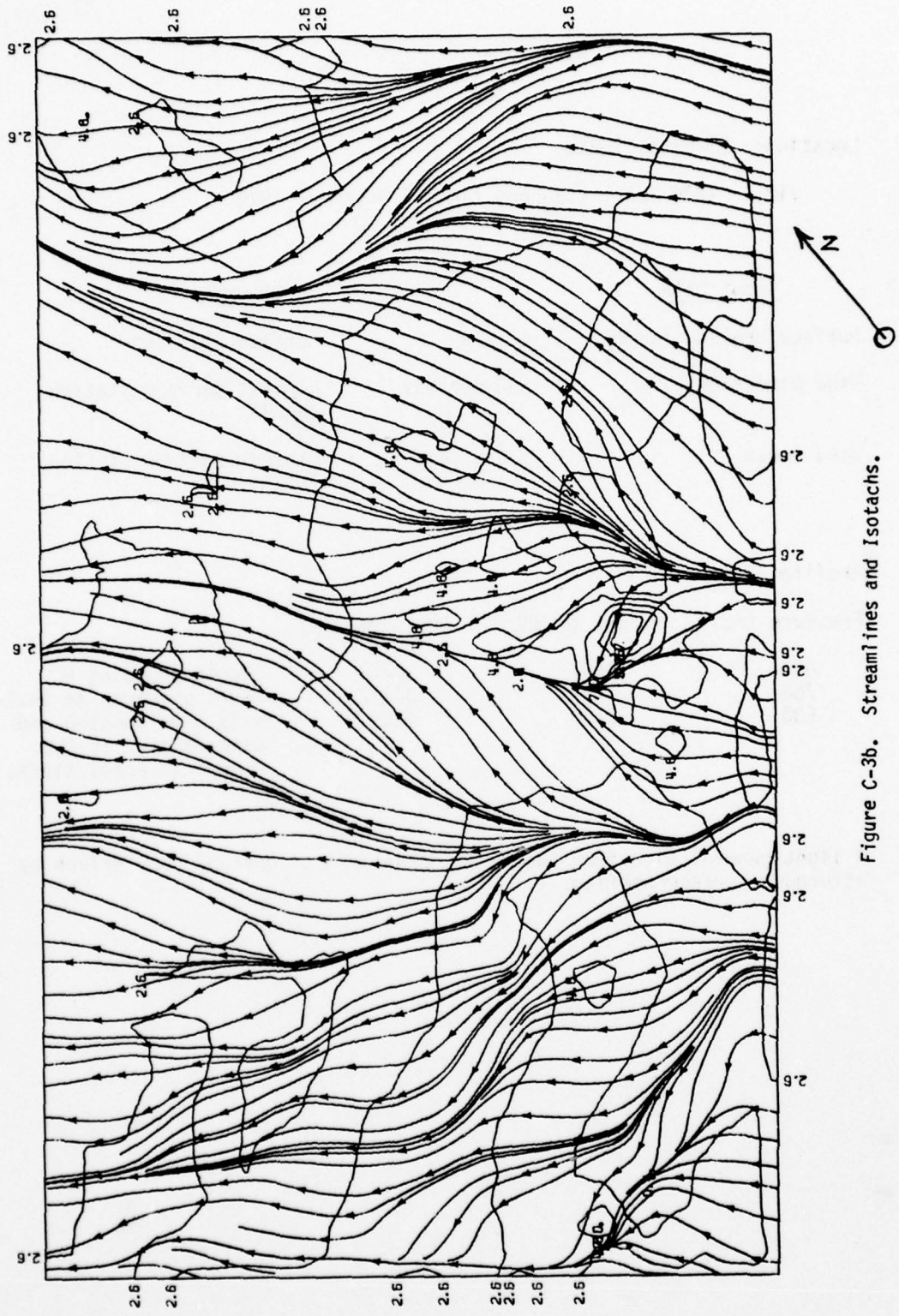


Figure C-3b. Streamlines and Isotachs.

FIGURE C-4

Location: El Paso, Texas

Time: 1400 local standard time, 1 November 1974

Input Data		Source
Surface Heating ($\Delta Q/c_p$)	+2.7 K	Surface stations
Wind Direction	230 degrees	Closest surface station (HAFB)
Wind Speed	3.0 m/sec.	Closest surface station (HAFB)

Profiles:

Pressure (mb)	Height (m,MSL)	Pot. Temp (K)	
850	1497	302.7	Single sounding 9 hours previous to analysis time updated and extrapolated by GWC numerical predictions.
700	3102	309.2	
500	5786	325.7	

A light general wind from the SW in conditions of upslope flow driven by afternoon surface heating.

DRIVING WINDS: VE= 2.3 VN= 1.9
BUOY = 2.7 TER # = 14 WND # = 13

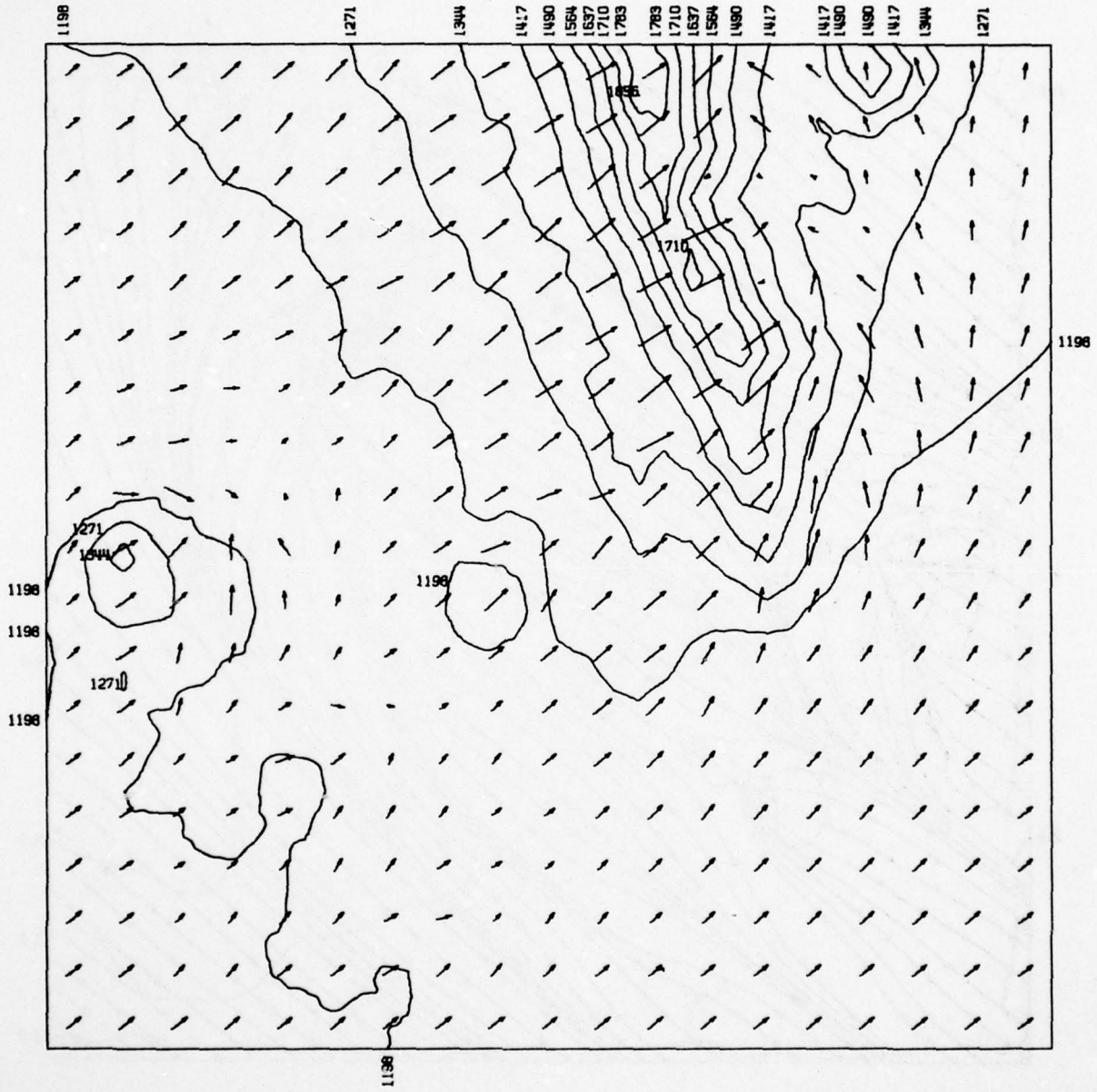


Figure C-4a. Terrain Contours and Wind Vectors.

DRIVING WINDS: VE= 2.3 VN= 1.9
BUOY = 2.7 TER # = 14 WND # = 13

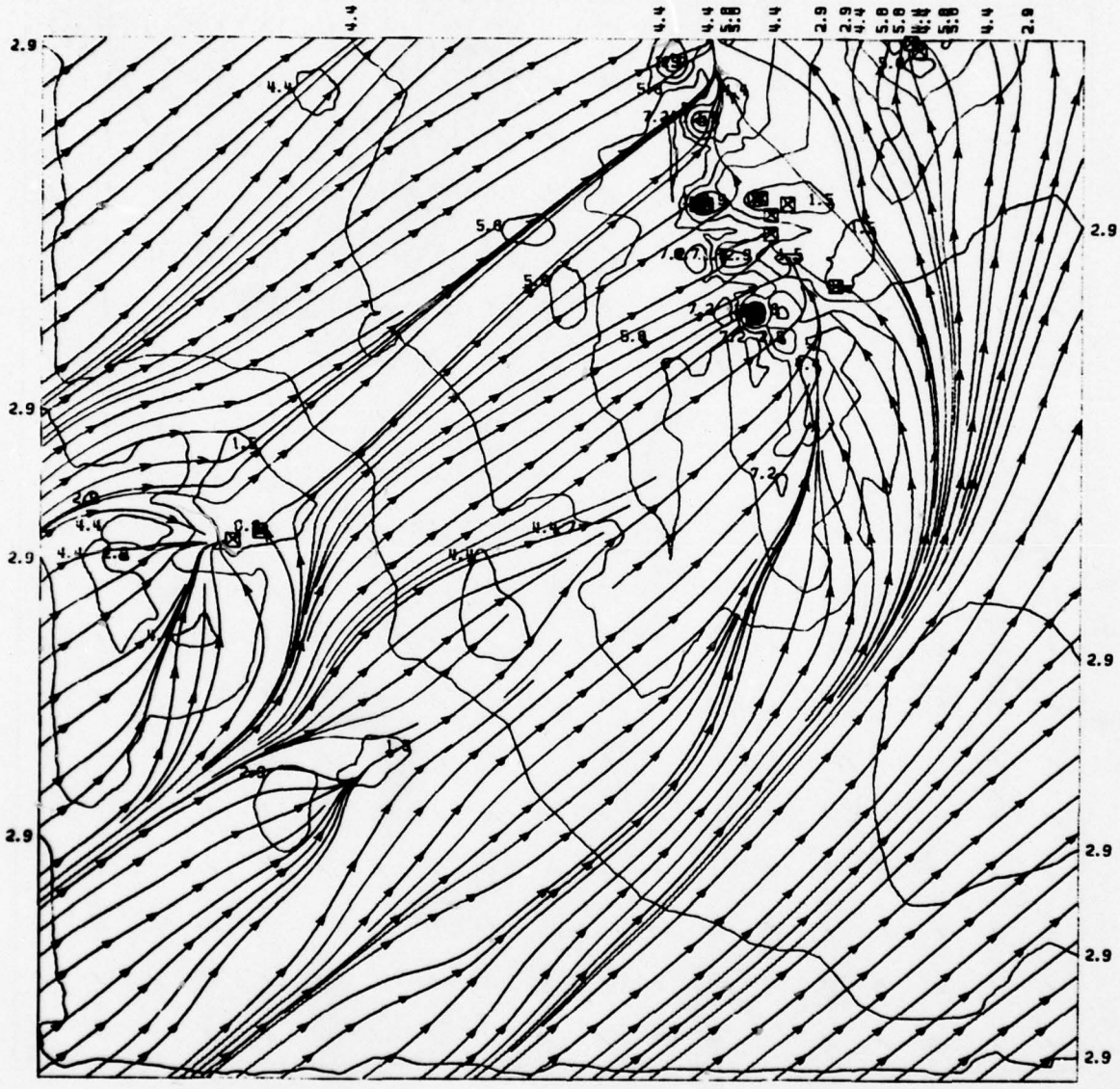


Figure C-4b. Streamlines and Isotachs.

FIGURE C-5

Location: El Paso, Texas

Time: 1400 local standard time, 3 November 1974

Input Data		Source
Surface Heating ($\Delta Q/c_p$)	+3.9 K	Surface stations
Wind Direction	320 degrees	Closest surface station (Juarez)
Wind Speed	7.7 m/sec	Closest surface station (Juarez)

Profiles:

Pressure (mb)	Height (m,MSL)	Pot. Temp (K)	
850	1472	295.8	Single sounding 9 hours previous to analysis time updated and extrapolated by GWC numerical predictions
700	3053	302.8	
500	5692	321.5	

Strong wind from the NW in conditions of moderate surface heating in mid-afternoon.

DRIVING WINDS: VE= 4.9 VN= -5.9
BUOY = 3.9 TER # = 14 WND # = 15

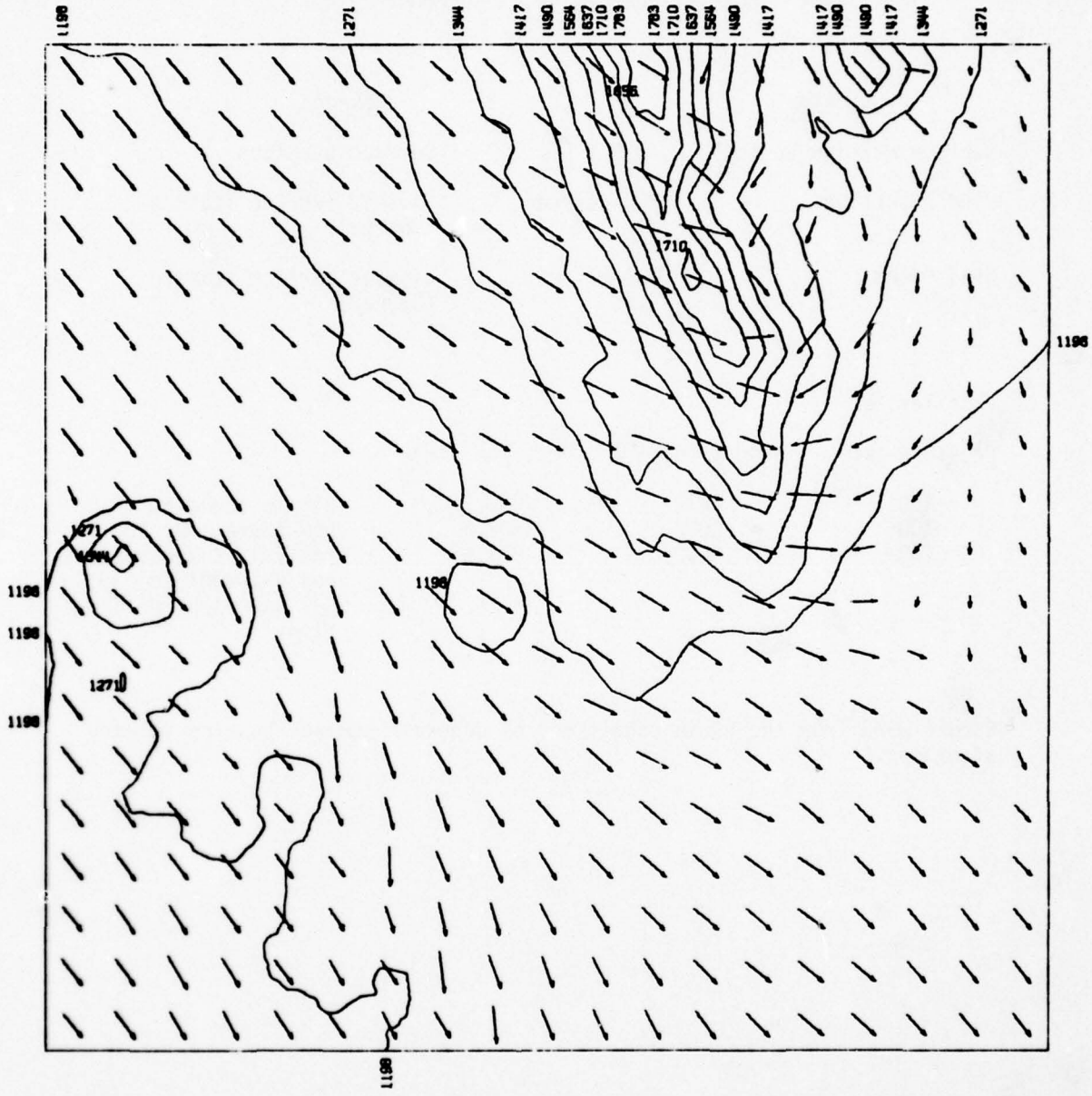


Figure C-5a. Terrain Contours and Wind Vectors.

DRIVING WINDS: VE= 4.9 VN= -5.9
BUOY = 3.9 TER # = 14 WND # = 15

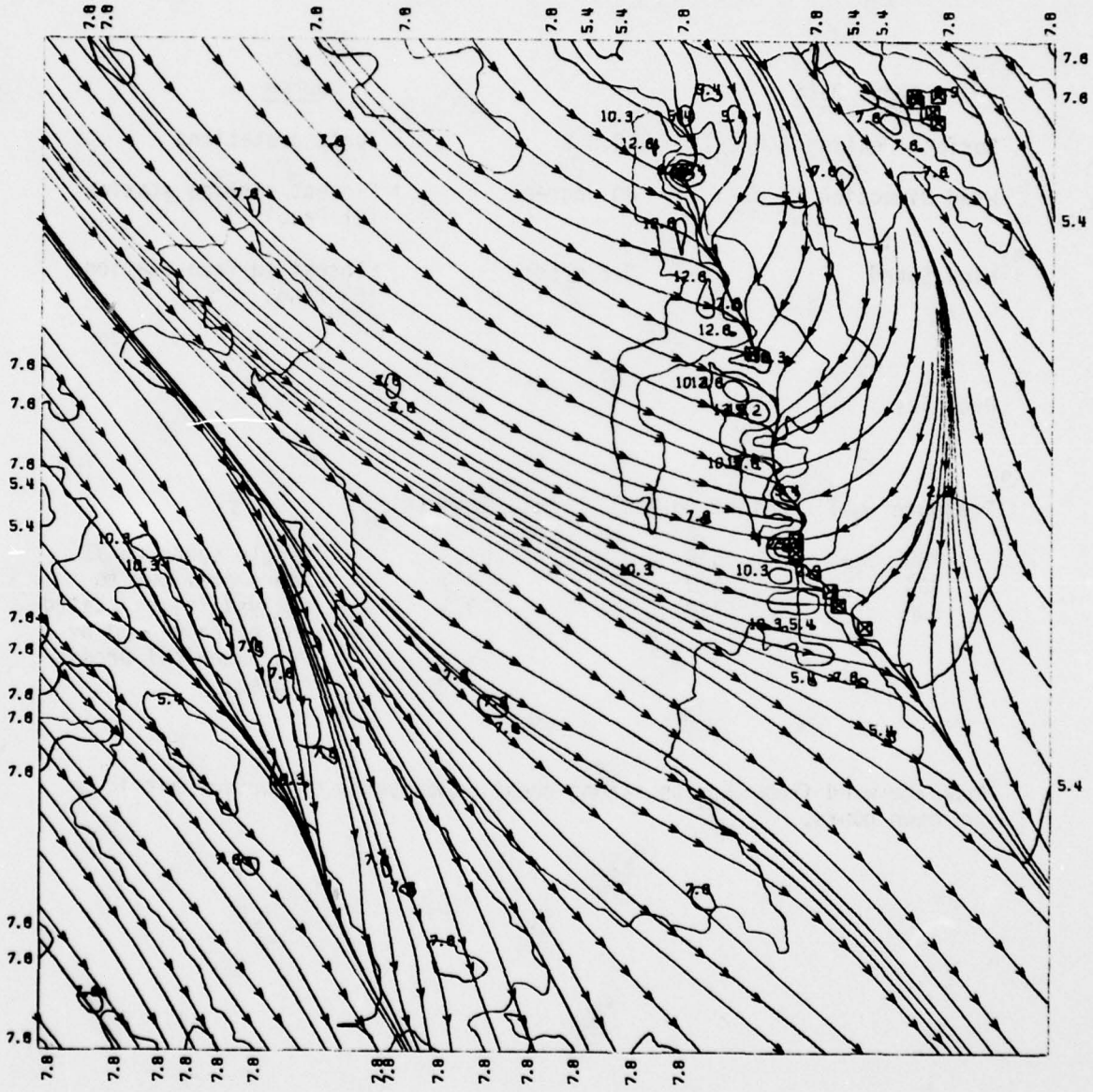


Figure C-5b. Streamlines and Isotachs.

FIGURE C-6

Location: El Paso, Texas

Time: 0400 local standard time, 5 November 1974

Input Data		Source
Surface Heating ($\Delta Q/c_p$)	-9.9 K	Surface stations
Wind Direction	40 degrees	Closest surface station (El Paso)
Wind Speed	3.6 m/sec.	Closest surface station (El Paso)

Profiles:

Pressure (mb)	Height (m,MSL)	Pot. Temp (K)	
850	1538	297.8	Single sounding 11 hours previous to analysis time updated and extrapolated by GWC numerical predictions.
700	3099	303.9	
500	5730	319.5	

Moderate wind from NE with strong negative buoyancy of surface air in pre-dawn hours.

DRIVING WINDS: VE= -2.3 VN= -2.8
BUOY = -9.9 TER # = 14 WND # = 12

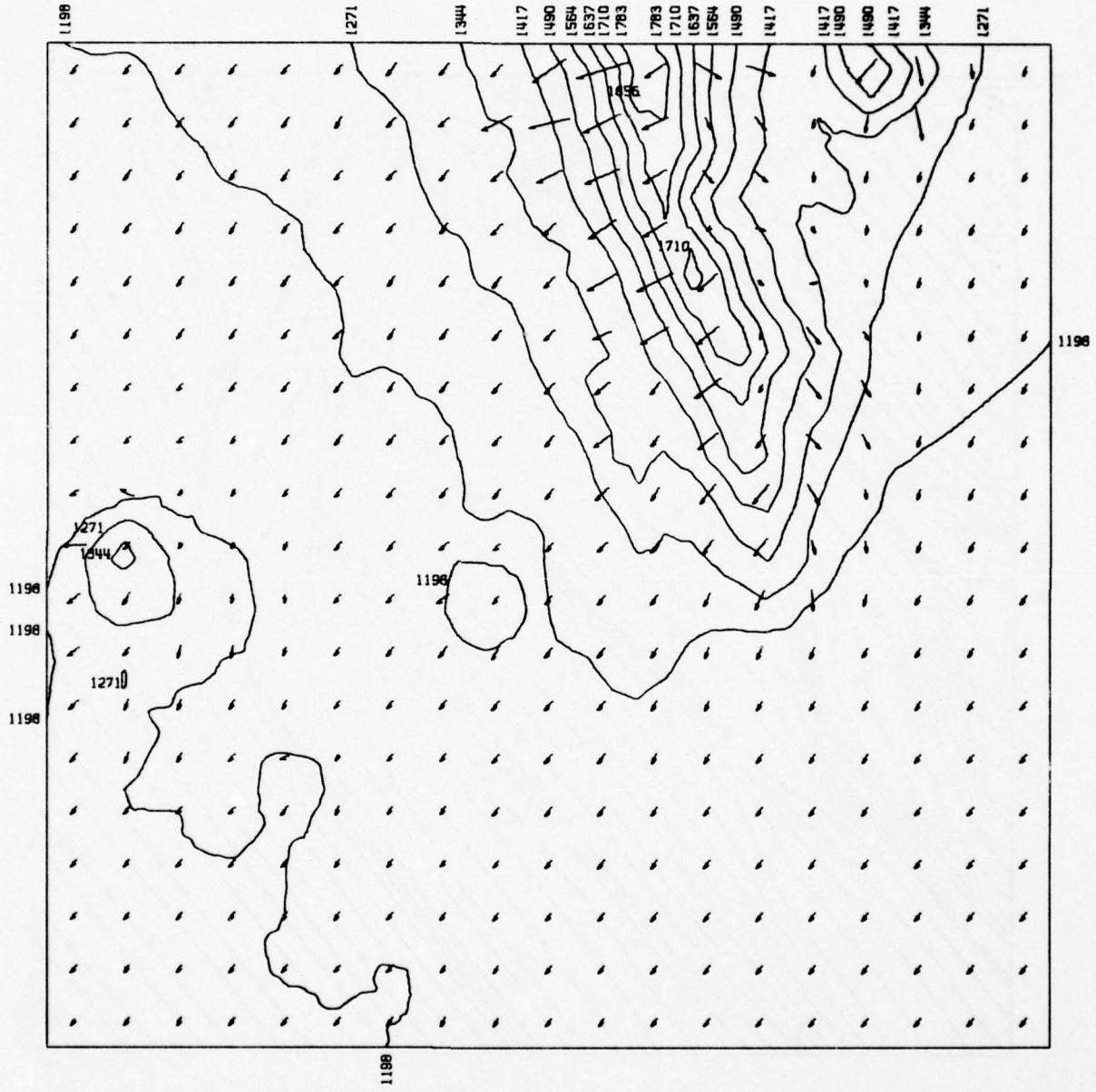


Figure C-6a. Terrain Contours and Wind Vectors.

DRIVING WINDS: VE= -2.3 VN= -2.8
BUOY = -9.9 TER # = 14 WND # = 12

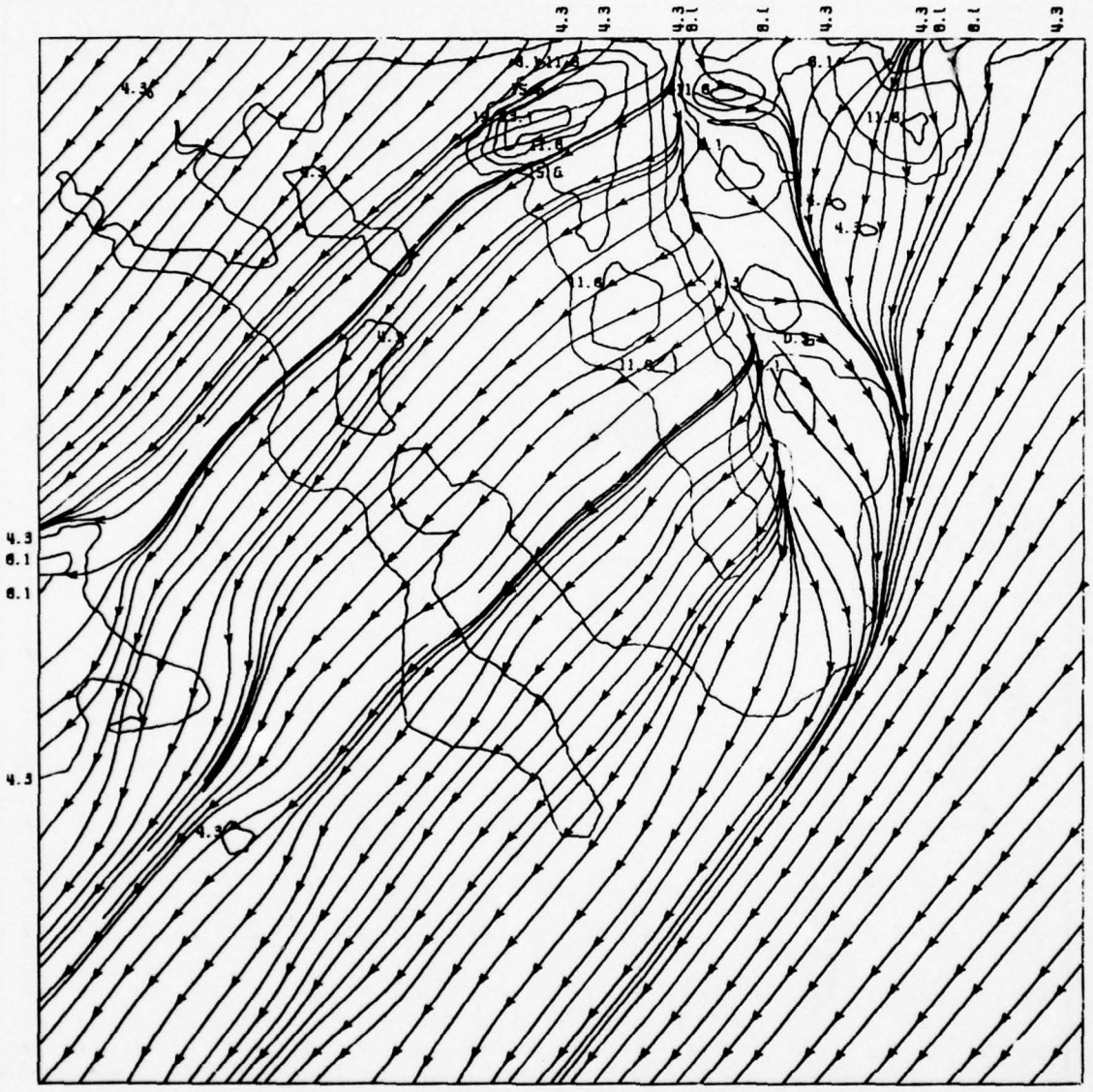


Figure C-6b. Streamlines and Isotachs.

FIGURE C-7

Location: El Paso, Texas

Time: 0400 local standard time, 2 November 1974

Input Data		Source
Surface Heating ($\Delta Q/c_p$)	-10.5 K	Surface stations
Wind Direction	N 35 degrees	Geostrophic wind model
Wind Speed	N 7 m/sec.	Geostrophic wind model

Profiles:

Pressure (mb)	Height (m,MSL)	Pot. Temp (K)	
850	1477	302.7	*Single sounding 11 hours* previous to analysis time updated and extrapolated by GWC numerical predictions
700	3099	307.4	
500	5762	321.9	

Strong wind from NE with strong negative buoyancy of surface air in pre-dawn hours.

DRIVING WINDS: VE= -3.9 VN= -5.8
BUOY = -10.5 TER # = 14 WND # = 9

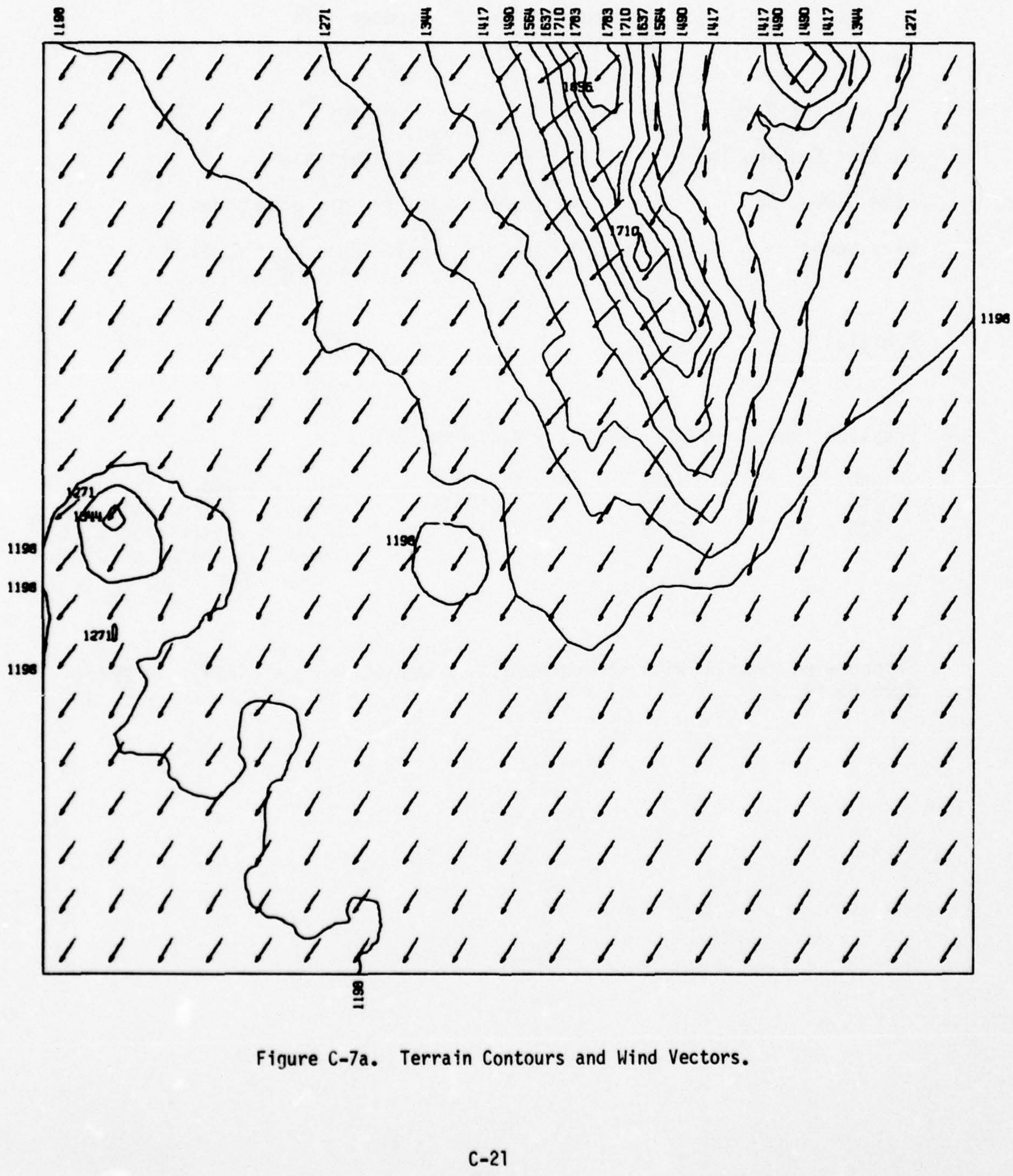


Figure C-7a. Terrain Contours and Wind Vectors.

DRIVING WINDS: VE= -3.9 VN= -5.8
BUOY = -10.5 TER # = 14 WND # = 9

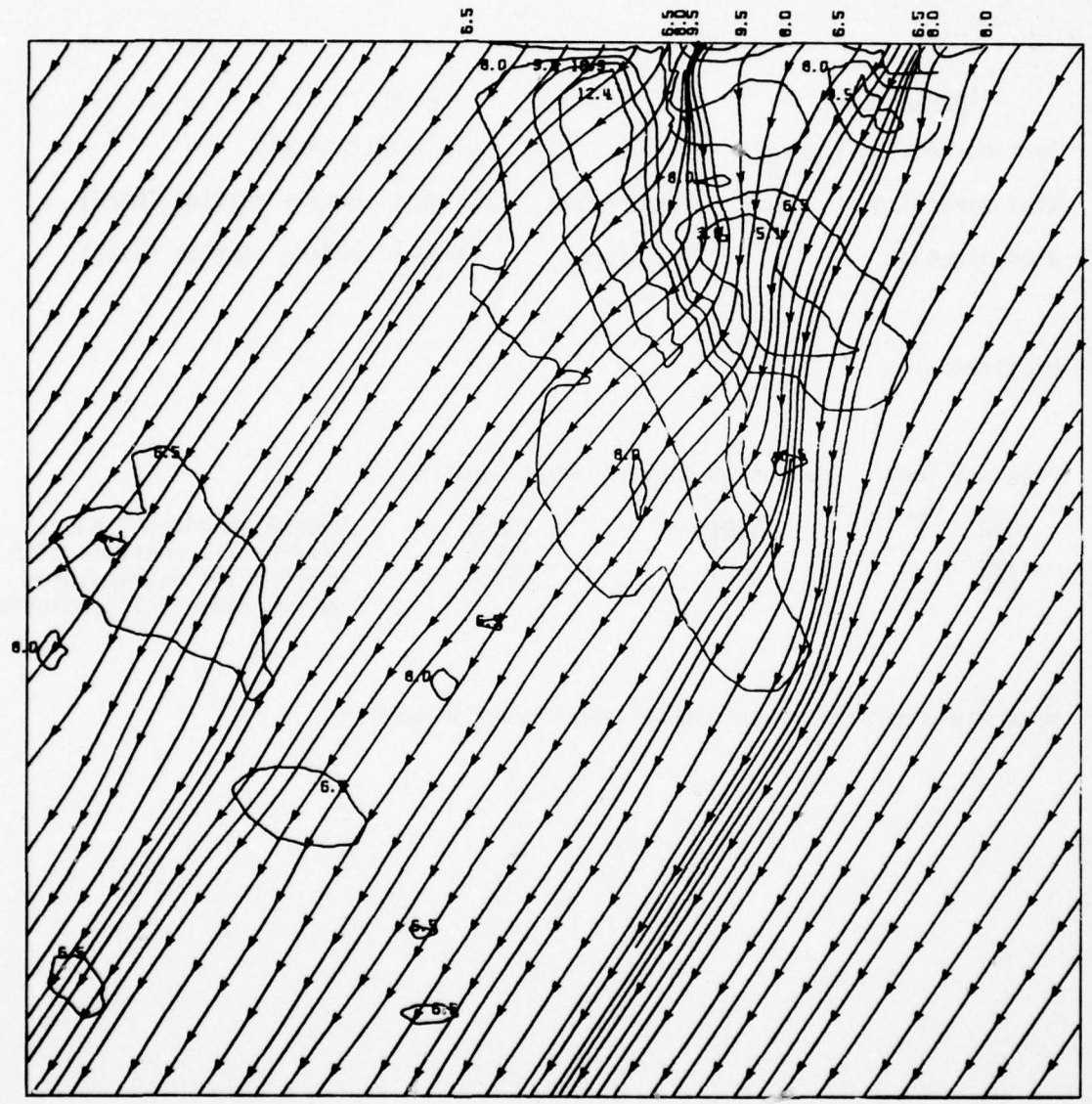


Figure C-7b. Streamlines and Isotachs.

FIGURE C-8

Location: Bear Canyon, WSMR

Time: 1400 local standard time, 2 November 1974

Input Data		Source
Surface Heating ($\Delta Q/c_p$)	+6.0 K	Surface stations
Wind Direction	190 degrees	Closest surface station (HAFB)
Wind Speed	5.1 m/sec.	Closest surface station (HAFB)

Profiles:

Pressure (mb)	Height (m,MSL)	Pot. Temp (K)	
850	1504	300.0	Single sounding 9 hours previous to analysis time updated and extrapolated by GWC numerical predictions.
700	3109	305.8	
500	5767	323.5	

Moderate wind from south with strong afternoon heating.

DRIVING WINDS: VE= .9 VN= 5.0
BUOY = 6.0 TER # = 11 WND # = 7

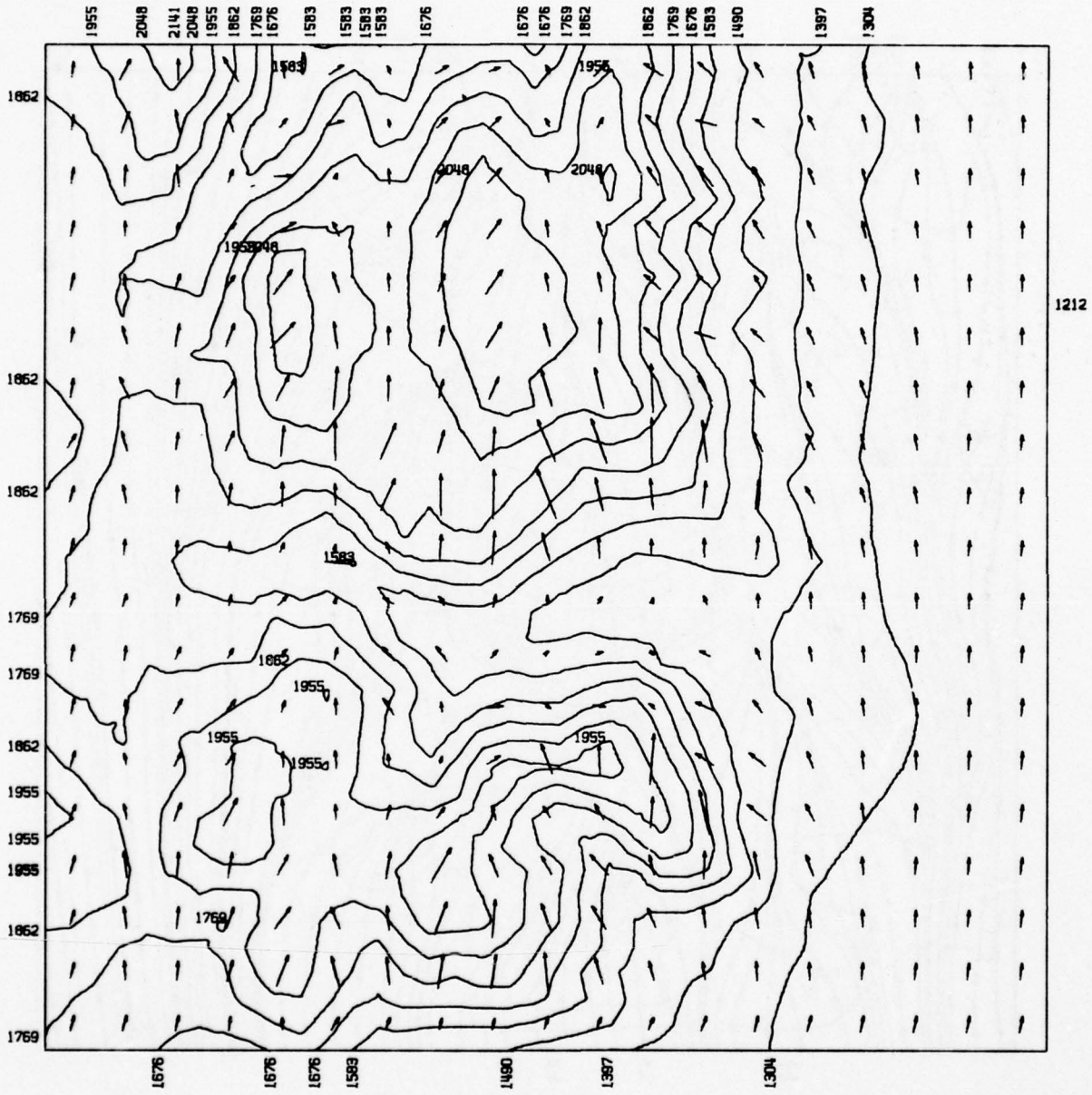


Figure C-3a. Terrain Contours and Wind Vectors.

DRIVING WINDS: VE= .9 VN= 5.0
BUOY = 6.0 TER # = 11 WND # = 7

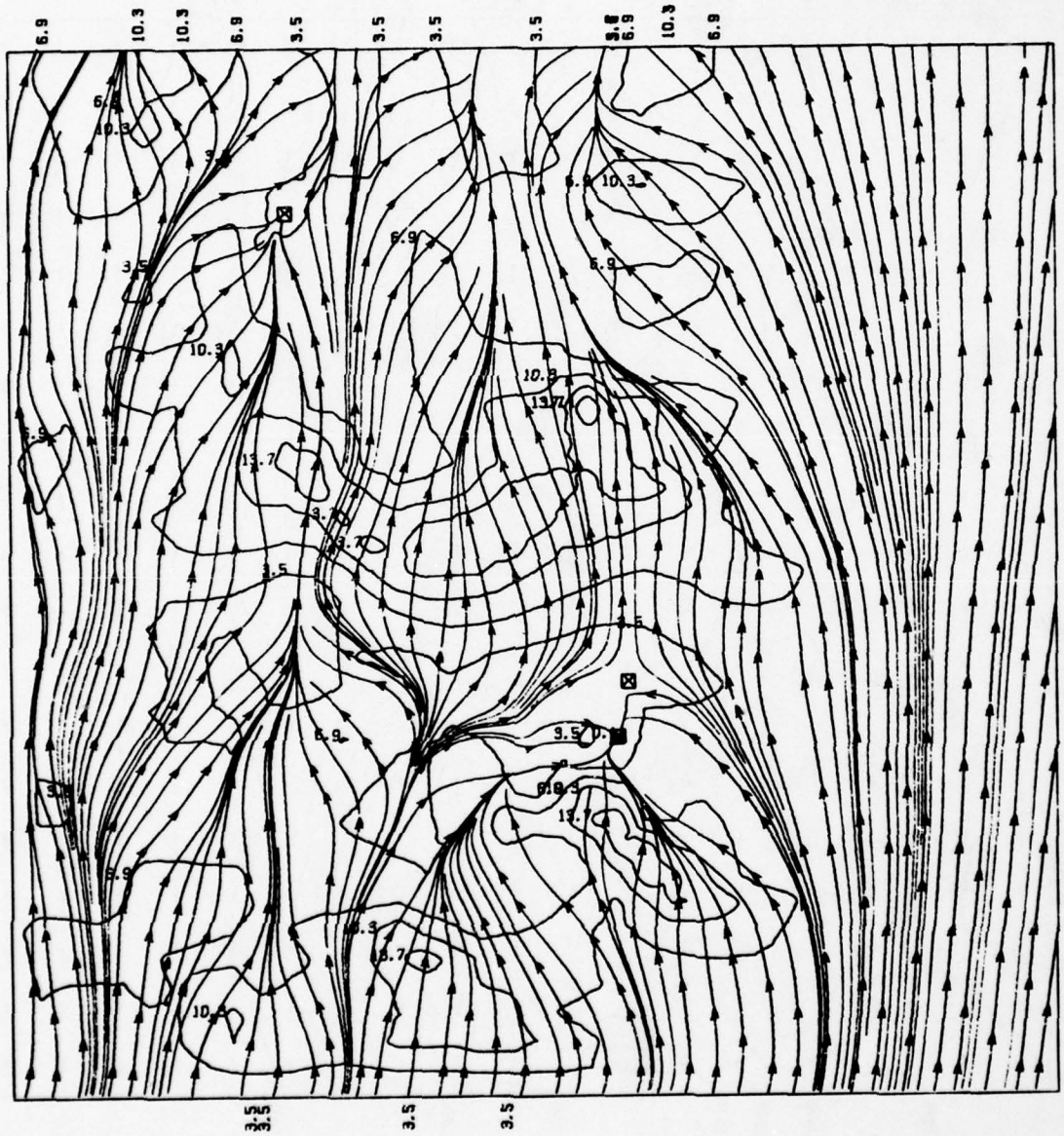


Figure C-8b. Streamlines and Isotachs.

FIGURE C-9

Location: Soledad Canyon, WSMR

Time: 1400 local standard time, 2 November 1974

Input Data		Source
Surface Heating ($\Delta Q/c_p$)	+6.0 K	Surface stations
Wind Direction	230 degrees	Closest surface station (El Paso)
Wind Speed	2.5 m/sec.	Closest surface station (El Paso)

Profiles:

Pressure (mb)	Height (m,MSL)	Pot. Temp (K)	
850	*1504*	*300.0*	*Single sounding 9 hours*
700	3109	305.8	previous to analysis time
500	5767	323.5	updated and extrapolated by GWC numerical predictions

Moderate wind from SW with strong afternoon heating.

DRIVING WINDS: VE= 1.9 VN= 1.6
BUOY = 6.0 TER # = 10 WND # = 6

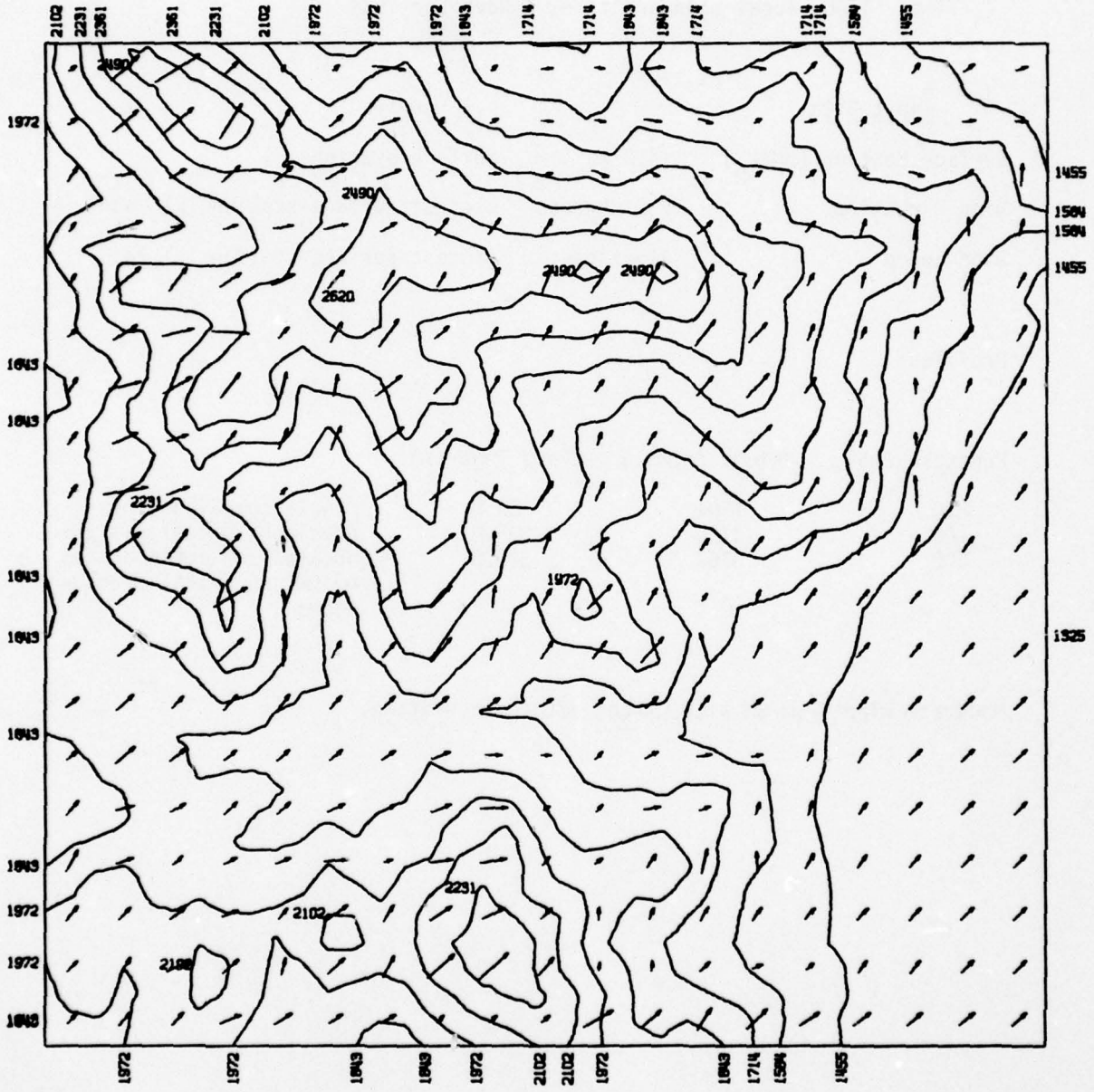


Figure C-9a. Terrain Contours and Wind Vectors.

DRIVING WINDS: VE= 1.9 VN= 1.6
BUOY = 6.0 TER # = 10 WND # = 6

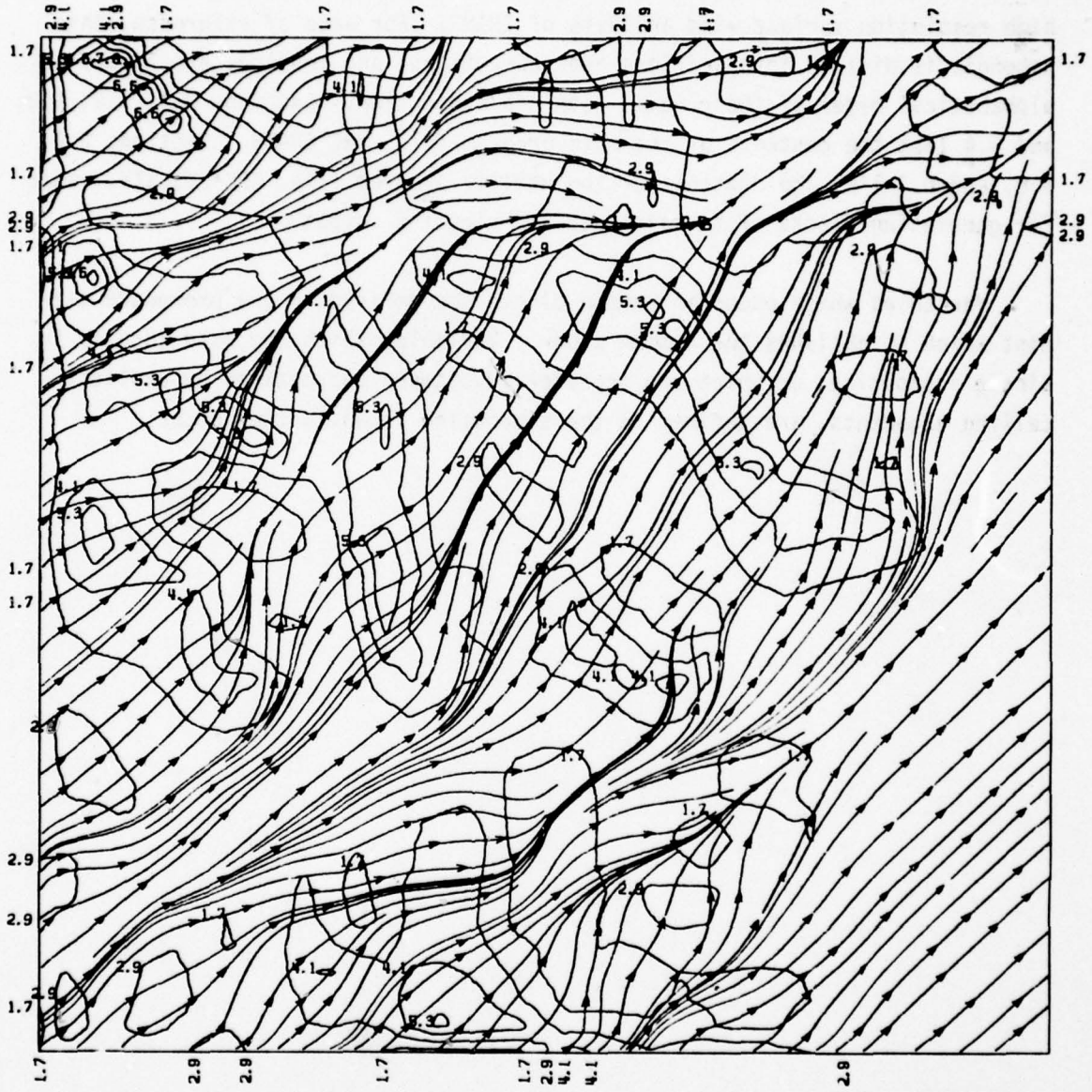


Figure C-9b. Streamlines and Isotachs.

APPENDIX D

FORTRAN LISTING OF COMPUTER PROGRAMS

This appendix contains a complete listing of all program elements and source routines, with the exception of ASL library routines, contained in the high resolution surface wind analysis of EPAMS. For ease of reference, this appendix is divided into sections and subsections, and routines are listed in alphabetical order of their names within a subsection. Sections D.1, D.2, D.3, and D.4 list the contents of the main program file, MRC*EPAMS. Sections D.5, D.6, and D.7 list the contents of the utility program file, MRC*UTILITY., used for generating blocks of terrain data and plotting output.

Variables which occur in common blocks are defined in the procedure element which establishes the common block. Variables which are used locally in single subroutine, or which are transferred between subroutines by explicit calling arguments, are defined in the subroutine in which they occur.

D.1 USERGUIDE

Contained within MRC*EPAMS is the element M.USERGUIDE which may be listed to obtain a brief description of the surface wind analysis and instructions for its use. Input variables and control flags for the overall operation of the wind analysis are defined in this element. In addition, control flags for the utility plotting routines are defined.

INTRODUCTION

THE MISSION RESEARCH CORPORATION (MRC) MICRO WIND MODEL IS A PHYSICALLY BASED NUMERICAL MODEL DESIGNED TO ESTIMATE SURFACE WINDS OVER A LIMITED AREA TAKING INTO ACCOUNT BOTH RUGGED TOPOGRAPHY AND THERMAL STRUCTURE. THIS MODEL IS INCORPORATED IN A COMPUTER PROGRAM DELIVERED TO THE WSMR ATMOSPHERIC SCIENCE LAB (ASL). THE ASL PROGRAM PERFORMS THREE MAIN FUNCTIONS:

1. ACCESS OF ASL METEOROLOGICAL AND TOPOGRAPHICAL DATA BASES
2. ESTIMATION OF MESOSCALE WIND AND TEMPERATURE CONDITIONS
3. ESTIMATION OF SURFACE WINDS USING THE MRC WIND MODEL

COMPLETE DOCUMENTATION OF THE MODEL AND COMPUTER PROGRAM MAY BE FOUND IN THE FINAL REPORT FOR THIS CONTRACT (NO. DAEA 18-177-C-0043). THE FOLLOWING IS A GUIDE TO RUNNING THE COMPUTER PROGRAM ON THE WSMR UNIVAC 1108 FACILITIES.

MAIN PROGRAM, UTILITY, AND DATA FILES.

SOFTWARE CODE AND STRUCTURE DELIVERED TO THE ASL CONSISTS OF A MAIN PROGRAM FILE (MRC*EPAMS.; HEREAFTER REFERRED TO AS M.), A UTILITY PROGRAM FILE (MRC*UTILITY.; HEREAFTER REFERRED TO AS MT.), AND FOUR DATA FILES, ALL STORED ON UNIVAC 1108 LIBRARY TAPE. BOTH PROGRAM FILES CONTAIN THE FOLLOWING TYPES OF ELEMENTS: FORTRAN SOURCE, RELOCATABLE, ABSOLUTE, JOB CONTROL, AND SAMPLE INPUT DATA. M. IS USED FOR THE CREATION AND EXECUTION OF THE SURFACE WIND MODEL, WHILE MT. SUPPLIES PLOTTING TERRAIN FILTERING, AND FILE MANAGEMENT ROUTINES.
THE DATA FILES ARE:
SWUS*MRCTRF.
SWUSMRC*WNDARCHIVE.
EURO*MRCTRF.
EUROMRC*WNDARCHIVE.
THE DATA FILES CONTAIN THE MICRO TERRAIN INPUT AND MICRO WIND OUTPUT FOR THE SOUTHWEST U.S. AND EUROPE.

HOW TO RUN THE SURFACE WIND PROGRAM

THE FOLLOWING STEPS MAY BE TAKEN,USING A DEMAND TERMINAL, TO EXECUTE THE SURFACE WIND PROGRAM IN BATCH MODE:

1. ASSIGN M.
2. SET UP INPUT FILE
3. EXECUTE

1. TO ASSIGN PROGRAM FILE M., TYPE IN

```
@USE M.,MRC*EPAMS.
@ASG,AZ M.
```

IF M. IS DECATALOGUED, IT MUST BE COPIED IN FROM TAPE. GET HELP IF NECESSARY.

2. ALL INPUT IS READ VIA NAMELIST FORMAT FROM FILE XINFIL.. PRELIMINARY PREPARATION OF THIS FILE MAY BE ACHIEVED BY TYPING IN

```
@ADD M.BATCHINP
```

THIS CREATES THE INPUT FILE AND CALLS THE ED PROCESSOR. THE USER MAY THEN EDIT XINFIL. AS DESIRED.

A DESCRIPTION OF THE INPUT VARIABLES FOLLOWS.

NAMELIST /XFLAGS/ - THESE FLAGS ARE USED BY THE EPAMS EXECUTIVE TO CONTROL PROGRAM FLOW. TO EXECUTE THE MRC WIND MODEL. FLAGS SHOULD BE SET AS FOLLOWS:

FUNCTION = 2
TASK = 2
JOB = 0

NAMELIST /FLAGS/ - THESE FLAGS CONTROL THE DATA BASE SEARCH AND THE MICRO WIND SIMULATION.

IDAT(1) = 0 SEARCH UPPER AIR DATA INVENTORY
= 1 SKIP UA DATA

IDAT(2) = 0 SEARCH SURFACE STATION DATA INVENTORY
= 1 SKIP SS DATA

IDAT(3) = 0 SEARCH GWC DATA INVENTORY
= 1 SKIP GWC DATA
NO GWC DATA EXISTS FOR EUROPEAN VERSION

IDAT(4) = 0 USE MESOMODEL DATA
= 1 SKIP MESOMODEL DATA
MESOMODEL DATA REQUIRES SPECIAL HANDLING

IDAT(5) = 0 USE MESONET DATA
= 1 SKIP MESONET DATA
MESONET DATA NOT PRESENTLY AVAILABLE

IDAT(6) = 0 USE MANUAL INPUT
= 1 SKIP MANUAL INPUT

IMDT(1) - UPPER AIR TIME INTERPOLATION FLAG
= 0 SEEK UA OBSERVATION JUST PRECEDING SIMULATION TIME
= 1 SEEK TWO UA OBSERVATIONS CLOSEST TO SIMULATION TIME AND LINEARLY INTERPOLATE
= 1 RECOMMENDED FOR EUROPEAN VERSION

IMDT(2) - SURFACE STATION TIME CRITERIA FLAG.
SS DATA LESS THAN OR EQUAL TO IMDT(2) HOURS OLD WILL BE SOUGHT.
= 2 RECOMMENDED

IMDT(3) - GWC FLAG (NOT USED)

IMDT(4) - MESOMODEL DATA FLAG
MUST BE SET EQUAL TO 0

IMDT(5) - MESONET FLAG (NOT USED)

IMDT(6) - MANUAL INPUT FLAG (NOT USED)

ILASTR - MULTIPLE RUN FLAG
= 0 WNDMGR WILL CARRY OUT THIS RUN AND THEN READ FLAGS FOR A SUBSEQUENT RUN
= 1 WNDMGR WILL CARRY OUT THIS RUN AND THEN RETURN CONTROL TO THE EPAMS EXECUTIVE
= 1 REQUIRED FOR SINGLE RUNS

NTST - REQUIRED-ARCHIVING FLAG
- IF=1, RESULTS ARE ARCHIVED IF A RUN IS MADE

IFLUX - NUMBER OF FLUX BOXES USED IN WINDEX
MUST BE 2 OR 4
= 2 RECOMMENDED

```
      IHIERC            -    MODEL COMPLEXITY FLAG
                        = 0  DATA BASES ARE SEARCHED. MESOSCALE
                             DRIVERS ARE ESTIMATED. BUT WINDEX IS
                             NOT EXECUTED.
                        = 1  WINDEX IS EXECUTED. WINDFIELDS ARE
                             RELAXED. BUT TEMPERATURE FIELD IS FIXED
                        = 2  WIND AND TEMPERATURE FIELDS ARE RELAXED
      MTRBLK            -    MICRO TERRAIN DATA ACQUISITION FLAG.
                             IF MTRBLK.LE.0. THE MICRO TERRAIN
                             HEADER IS SEARCHED FOR A BLOCK OF TERRAIN
                             WITH SOUTHWEST CORNER WITHIN 10 KM OF
                             XPLACE.YPLACE (SEE BELOW).  IF MTRBLK.GT.0.
                             THEN MTRBLK SPECIFIES THE ARCHIVE NUMBER OF
                             THE TERRAIN DATA TO BE USED; XPLACE.YPLACE
                             ARE REDEFINED TO AGREE WITH THE SOUTHWEST
                             UTM COORDINATES OF THE ARCHIVED TERRAIN DATA
                             USED.
      IRELAX            -    NUMBER OF RELAXATION PASSES TO BE EXECUTED
                             IN WINDEX.  SHOULD BE GREATER THAN THE
                             MAXIMUM ARRAY DIMENSION
                        = 50 RECOMMENDED
      ILE               -    EASTERNMOST GRID INDEX LIMIT
      JLN               -    NORTHERNMOST GRID INDEX LIMIT
      ILW               -    WESTERNMOST GRID INDEX LIMIT
      JLS               -    SOUTHERNMOST GRID INDEX LIMIT
                        -    THE VARIABLES ILE.JLN.ILW.JLS MAY BE
                             USED TO RESTRICT THE REGION OVER WHICH
                             THE ANALYSIS IS PERFORMED.  MOSTLY
                             FOR USE IN DEBUGGING.
                            RECOMMENDED:  ILE = 40 SWUS     = 51  EUROPE
                                          JLN = 40 SWUS     = 31  EUROPE
                                          ILW =  1
                                          JLS =  1

NAMELIST /TIMPLA/       -    THESE FLAGS SPECIFY THE TIME AND PLACE OF
                             THE MICRO WIND SIMULATION
      XPLACE            -    UTM X-COORDINATE OF SOUTHWEST CORNER OF
                             SIMULATION SITE  (KM)
      YPLACE            -    UTM Y-COORDINATE OF SOUTHWEST CORNER OF
                             SIMULATION SITE  (KM)
                             XPLACE.YPLACE DO NOT HAVE TO BE SUPPLIED
                             IF MTRBLK.GT.0
      ZPLACE            -    ELEVATION OF SITE
                             SUPPLIED BY PROGRAM IF ENTERED AS 0.0
      YEAR              -      LOCAL
      MONTH             -       TIME
      DAY               -        OF
      HOUR              -         SIMULATION
      MINUTE            -    NOT USED

      THE MODEL MAY BE RUN INDEPENDENTLY OF THE DATA BASE SEARCH AND
      ANALYSIS BY MANUALLY ENTERING THE DRIVING WIND AND TEMPERATURE
      DATA.  TO PROPERLY SET UP THE INPUT FILE. TYPE IN THE FOLLOWING:
          @ADD M.BATCHINP
          ADD M.MANINP
```

AT THIS POINT, XINFIL. WILL CONTAIN THE USUAL NAMELISTS PLUS THE NAMELIST /MFLAGS/. ALL VARIABLES WILL HAVE STANDARD VALUES. THE USER MAY EDIT THESE AS DESIRED. THE ADDITIONAL FLAGS IN NAMELIST /MFLAGS/ ARE DESCRIBED BELOW:

NAMELIST /MFLAGS/

TPRO(1)	-	POTENTIAL
TPRO(2	-	
TPRO(3	-	TEMPERATURE
ZPRO(1)	-	
ZPRO(2)	-	PROFILE
ZPRO(3)	-	
DELQ	-	SURFACE HEATING--APPROXIMATELY THE SAME NUMERICAL VALUE AS (SURFACE TEMPERATURE)-(AMBIENT TEMPERATURE)
GENWND(1)	-	EASTERLY COMPONENT OF MESOSCALE WIND
GENWND(2)	-	WESTERLY COMPONENT OF MESOSCALE WIND
NUMOBS	-	NUMBER OF SPOT OBSERVATIONS. IF NUMOBS.GT.0, $MFLAGS INPUT MUST BE FOLLOWED BY NUMOBS LINES OF INPUT, FORMATTED I5,I5,F10.2,F10.2,F10.2 LINE # I GIVES DATA FOR THE FOLLOWING VARIABLES:
KINDX(I)	-	LOCATION INDICES FOR
LINDX(I)	-	SPOT OBSERVATION # I
THETA0(I)	-	POTENTIAL TEMPERATURE OF SPOT OBSERVATION # I
UW(I)	-	EASTERLY WIND COMPONENT OF SPOT OBSERVATION # I
VW(I	-	NORTHERLY WIND COMPONENT OF SPOT OBSERVATION # I

3. TO EXECUTE, TYPE IN

```
@ADD M.GOBATCH/SWUS
@START,/R  M.BATCH,,XXXXXX.8100,,[TIMEST],[PAGEST]
```

THE ESTIMATED WIND AND TEMPERATURE FIELDS WILL BE ARCHIVED IN FILE SWUSMRC*WNDARCHIVE (OR EUROMRC*WNDARCHIVE)

HOW TO GET PLOTS

BATCH PLOT JOBS MAY BE SUBMITTED FROM DEMAND TERMINALS IN A MANNER SIMILAR TO THE ABOVE: ASSIGN MT., SET UP INPUT FILE, EXECUTE.

1. TO ASSIGN MT., TYPE IN

```
@USE MT.,MRC*UTILITY.
@ASG,AZ  MT.
```

2. TO SET UP THE INPUT FILE, TYPE IN

```
@ADD MT.PBATCHINP
```

THIS WILL CREATE XINFIL. WITH STANDARD VALUES GIVEN TO THE NAMELIST INPUTS; THE ED PROCESSOR IS CALLED: THE USER MAY EDIT THE INPUT FILE AS DESIRED. INPUT VARIABLES ARE DESCRIBED BELOW.

```
NAMELIST /OMPLOT/  -    THESE FLAGS ARE USED BY THE PLOT EXECUTIVE
                       TO CALL AND CONTROL ASL LIBRARY PLOTTING
                       ROUTINES
WATFIL             -    NUMBER OF FILE CONTAINING DATA TO BE
                       BLOCKED IN
                   = 17 WIND OR TEMPERATURE PLOTS
                   = 16 TERRAIN PLOTS
ISTASH             -    ARCHIVE NUMBER OF DATA TO BE PLOTTED.
ILL                -    GRID PARAMETER
JLL                -    GRID PARAMETER
                       FOR SWUS.   SET ILL = 40
                                       JLL = 40
                       FOR EUROPE. SET ILL = 51
                                       JLL = 31
ILE                -      GRID INDEX LIMITS.
ILW                -      SHOULD BE
JLN                -      SAME AS VALUES
JLS                -      USED WITH WIND MODEL
ITYPE              -    SPECIFIES TYPE OF PLOT DESIRED
                   = 1  CONTOUR PLOT OF TERRAIN
                   = 2  CONTOUR PLOT OF TEMPERATURE
                   = 3  STREAMLINES OF WINDFIELD WITH SPEED
                       CONTOURS
                   = 4  VECTOR PLOT OF WINDFIELD WITH SCALED
                       MAGNITUDES
                   = 5  THREE DIMENSIONAL PLOT OF TERRAIN
NUMUP              -    NUMBER OF PLOTS TO BE VERTICALLY STACKED
                   = 1 RECOMMENDED
PLH                -    HEIGHT OF PLOT IN INCHES
                   =   25.0/NUMUP 15 MAXIMUM ALLOWABLE
SAMPLE             -    FOR PLOTTING EVERY OTHER DATA POINT
                   = 1  PLOT ALL POINTS
                   = 2  PLOT EVERY OTHER DATA POINT
                   = 2  RECOMMENDED FOR PLH LESS THAN 15.0
WATPEN             -    PEN COLOR
                   = 1  BLACK
                   = 2  RED
                   = 3  BLUE
                   = 1  RECOMMENDED FOR CONTOURS
                   = 3  RECOMMENDED FOR STREAMLINES OR VECTORS
OVRLAY             -    OVERLAY FLAG
                   = 0  PLOT IS NOT OVERLAID
                   = 1  PLOT IS LAID OVER PREVIOUS PLOT
                   = 1  RECOMMENDED FOR CONTOURS FOLLOWING
                       ITYPE=4 PLOTS.
NCON               -    NUMBER OF CONTOURS TO BE DRAWN
                   = 8  DEFAULT VALUE
TEXT               -    BCD ARRAY TO BE PLOTTED AS TITLE.
                       ALWAYS SUPPLIED A DEFAULT VALUE
                       AT PRESENT
ISTOP              -    STOPPING FLAG   USE IS OPTIONAL
                   = 1  NO PLOT IS EXECUTED. PROGRAM HALTS
```

3. *TO EXECUTE THE PLOT JOB, TYPE IN*
   @ADD MT.GOPBATCH

THE NUMBER OF THE PLOT TAPE CREATED WILL BE GIVEN ON THE PRINT OUTPUT.

D.2 PROCEDURE ELEMENTS OF MRC*EPAMS

This section contains relocatable elements, mostly labeled common blocks, of MRC*EPAMS. Variables are defined within the element in which they occur.

```
      COMMON /ABUFER/ A(5700)
 END
COEF    PROC
C
      PARAMETER NVAR=4, LEVELS=3
      COMMON /COEF/ AA(LEVELS,NVAR), BB(LEVELS,NVAR),
     1              CC(LEVELS,NVAR), IFLG1
C
C     THESE VARIABLES REPRESENT COEFFICIENTS OF PLANE
C
C
C               NVAR = NUMBER OF VARIABLES
C             LEVELS = NUMBER OF CONSTANT PRESSURE LEVELS
 END
COORD   PROC
      COMMON /COORD/ XPLACE, YPLACE, ZPLACE, YEAR,MONTH,DAY,HOUR,
     1  MINUTE, TIME, LTIME
      INTEGER YEAR, MONTH, DAY, HOUR, MINUTE, TIME
C     COORD IS COMMON STORAGE FOR THE TIME AND SPATIAL COORDINATES OF
C       THE MICRO WIND SIMULATION
C     XPLACE, YPLACE ARE THE UTM COORDINATES OF THE SOUTHWEST CORNER
C       OF THE REGION OF INTEREST IN KM

DATFLG  PROC
C
      COMMON /DATFLG/ NVFLAG(3)
C
C     THESE FLAGS MONITOR THE ACQUISITION OF WIND AND TEMPERATURE
C     INITIALIZATION DATA.
C
C     NVFLAG(1)       - 0:  GENWND NOT YET FOUND
C                       - 1:  FOUND
C     NVFLAG(2)       - 0:  SURFACE HEATING NOT YET FOUND
C                     - 1:  FOUND
C     NVFLAG(3)       - 0:  TEMPERATURE PROFILE NOT YET FOUND
C                     - 1:  FOUND
 END
FILES   PROC
C
      COMMON /FILES/ XINFIL,XOTFIL,OBFILE,UAFILE,GWCOBF,GWCFIL,
     1  SFCOBF,SFCFIL,RFILE,SPAREF,LYR2FL,LYR4FL,TRRNFL,
     2  MICTRF,MOTFIL,GRIDF1,GRIDF2,GRIDF3
      INTEGER XINFIL,XOTFIL,UAFILE,OBFILE,GWCOBF,GWCFIL,
     1  SFCOBF,SFCFIL,RFILE,SPAREF,LYR2FL,LYR4FL,TRRNFL,
     2  MICTRF,MOTFIL,GRIDF1,GRIDF2,GRIDF3
C
C **  THESE VARIABLES ARE PASSED TO THE 3 PHASES VIA THIS COMMON BLOCK
C
C     XINFIL - INPUT DATA (CARD) FILE
C     XOTFIL - OUTPUT (PRINT) FILE
C     OBFILE - UPPER AIR DATA MASTER DIRECTORY FILE
```

```
C     UAFILE - UPPER AIR DATA INVENTORY FILE
C     GWCOBF - GWC DATA MASTER DIRECTORY FILE
C     GWCFIL - GWC DATA INVENTORY FILE
C     SFCFIL - SURFACE DATA INVENTORY FILE
C     RFILE -
C     SPAREF - FILE USED TO STORE TEMPORY OUTPUT FROM ANALYSIS PHASE FOR
C              PRINTING AND/OR PLOTTING BY GRAPHIC PHASE
C     LYR2FL - HEC DATA FILE
C     LYR4FL - HEC DATA FILE
C     TRRNFL - MESOSCALE TERRAIN DATA FILE
C     MICTRF - MICRO TERRAIN FILE
C     MOTFIL - WINDEX SIMULATION OUTPUT FILE
C     GRIDF1 - MESO MODEL LAYER 1 OUTPUT FILE
C     GRIDF2 - MESO MODEL LAYER 2 OUTPUT FILE
C     GRIDF3 - MESO MODEL LAYER 3 OUTPUT FILE
C
 END
GBUFER  PROC
C
      COMMON /OBUFER/ GRIDX(PG),GRIDY(PG),GRDBLK(RG),ANLTIM(RG),RUNMAX,
     1  GRDMAX
      INTEGER GRDBLK,ANLTIM,RUNMAX,GRDMAX
C
C     GWC HEADER DATA IS READ INTO THIS BUFFER
C
C     GRIDX,GRIDY - UTM COORDINATES OF GWC GRID POINT
C     GRDBLK(I) - BLOCK NUMBER OF GWC FILE HOLDING DATA FOR RUN I
C     ANLTIM(I) - GWC ANALYSIS TIME FOR RUN I
C     RUNMAX - NUMBER OF GWC ANALYSIS TIMES IN DATA BASE
C     GRDMAX - NUMBER OF GWC GRID POINTS IN DATA BASE
C
 END
GWDATA  PROC
C
C     GWC PREDICTION DATA
C
C     GTIME - TIME
C     P - PRESSURE (PASCALS)
C     T - TEMP. DEG. K
C     Q - SPECIFIC HUMIDITY
C     U,V - WIND COMPONENTS (MPS)
C     Z - ELEVATION METERS
C
      COMMON /GWDATA/ GTIME,P(4),Q(4),T(4),U(4),V(4),Z(4)
      INTEGER GTIME
 END
INTLYZ  PROC
      PARAMETER NOBS=10
C
      COMMON /INTLYZ/ TPSITE, GENWND(2), TSITE, DELQ, DELTER
```

```
      COMMON /SURFOB/ KINDX(NOBS),LINDX(NOBS),UW(NOBS),
     1  VW(NOBS),THETA0(NOBS),NUMOBS
      COMMON /TPROF/ TPRO(3),ZPRO(3)
C
C     /INTLYZ/ IS USED TO PASS INITIALIZATION DATA TO SUBROUTINE WINDEX
C
C     TPSITE          - SURFACE POTENTIAL TEMPERATURE AT SIMULATION SITE
C     TSITE           - TEMPERATURE AT SITE
C     GENWND(1)       - EASTERLY COMPONENT OF BACKGROUND WIND
C     GENWND(2)       - NORTHERLY COMPONENT OF BACKGROUND WIND
C     DELQ            - MEASURE OF AVERAGE SURFACE HEATING
C     TPR(.)          -   POTENTIAL TEMPERATURE   (DEG K)
C     ZPR(.)          -   PROFILE AT SITE         (METERS)
C     DELTER          - TERRAIN GRID SPACING
 END
MESAGE  PROC
      COMMON /MESAGE/ FUNCTN, TASK, JOB, BLOCK, FILE, PRINT, LEVEL
      INTEGER FUNCTN, TASK, JOB, BLOCK, FILE, PRINT
C
C     THESE VARIABLES SPECIFY LOGICAL FLOW.
C     FUNCTN = 1 APPLICATION
C     FUNCTN = 2  ANALYSIS
C     FUNCTN = 4  PLOTTING
C     IF FUNCTN = 1, THEN TASK = 1 CALLS MIXLYR
C                          = 2 CALLS WNDMGR
C     JOB = 0  DEFAULT VALUE
C         = 1  DIAGNOSTIC
C         = 2  PREDICTION
C     BLOCK IS NOT YET DEFINED
C     FILE SPECIFIES OUTPUT DEBUG FILE FOR EXECUTION TIME DIAGNOSTICS
C     PRINT DETERMINES DEBUG OUTPUT
C     PRINT = 0  NO DEBUG OUTPUT
C           = 1  PRINT FORMAL ARGUMENT SUBROUTINE INPUT AND OUTPUT
C                  PARAMETERS
C           = 2  PRINT INTERMEDIATE RESULTS
C           = 3  PRINT ALL RELEVANT VARIABLES
C           = 4  PRINT DATA FILES WHEN BLKIN AND BLKOUT ARE CALLED
C
 END
MRCDAT  PROC
C
C     DTBMGR SEEKS INFORMATION FROM THE UPPER AIR, SURFACE, GWC,
C     MESOMODEL, AND MESONET DATA INVENTORIES.  THE INFORMATION MUST COME FROM
C     STATIONS WHICH ARE SUFFICIENTLY CLOSE IN TIME AND SPACE TO THE
C     SIMULATION TIME AND LOCATION.  THE INFORMATION IS STORED IN THESE
C     VARIABLES AND PASSED TO DATANL.
C
C
      PARAMETER NUAX=4, NLVL=25, NSFX=6, NGWX=4, ITT=2, NMMX=9,
     1   LAYERS=2
```

```
C     NUAX - NUMBER OF UA STATIONS SOUGHT
C     NLVL - NUMBER OF UA LEVELS SOUGHT
C     NSFX - NUMBER OF SURFACE STATIONS SOUGHT
C     NGWX - NUMBER OF GWC GRID POINTS SOUGHT
C     ITT - NUMBER OF GWC PREDICTION TIMES USED
C     NMMX - NUMBER OF MESO MODEL GRID POINTS SOUGHT
C     LAYERS - NUMBER OF MESO MODEL LAYERS SEARCHED FOR DATA
C
      COMMON/MRCDAT/NUANUM,STUNUM(NUAX),XSTA(NUAX),YSTA(NUAX),ZSTA(NU
     2 AX),  UDIST(NUAX),IFLAGU(NUAX),NTR(NUAX),UPTIME(NUAX),PRES(NLVL,
     3  NUAX),TEMP(NLVL,NUAX),QHUM(NLVL,NUAX),ZMSL(NLVL,NUAX),
     4  VRTEMP(NLVL,NUAX),POTEMP(NLVL,NUAX),UWIND(NLVL,NUAX),VWIND(NLVL
     5 ,NUAX),NWR(NUAX),UVEL(NLVL,NUAX),VVEL(NLVL,NUAX),ZWDMSL
     M  (NLVL,NUAX),NSFNUM,SDIST(NSFX),
     7  STSNUM(NSFX),XSTS(NSFX),YSTS(NSFX),ZSTS(NSFX),
     8  IFLAGS(NSFX),SFTIME(NSFX),WDSF(NSFX),WSSF(NSFX),WGSF(NSFX),
     9  SLPSF(NSFX),BTSF(NSFX),TSSF(NSFX),TDSF(NSFX),ASSF(NSFX),
     T  PASF(NSFX),TSCSF(NSFX),PWSF(NSFX),VVSF(NSFX),WWSF(4,NSFX),
     1  ICLOUD(9,NSFX),STDSTF(NSFX),NGWNUM(ITT),IGGNUM(ITT,NGWX),
     2  GDIST(ITT,NGWX),XGRD(ITT,NGWX),YGRD(ITT,NGWX),ZGRD(ITT,
     3  NGWX),IFLAGG(ITT,NGWX),GWTIME(ITT,NGWX),PGW(4,ITT,NGWX),
     4  QGW(4,ITT,NGWX),TGW(4,ITT,NGWX),UGW(4,ITT,NGWX),
     5  VGW(4,ITT,NGWX),ZZGW(4,ITT,NGWX),
     6  NMMNUM,RMDIST(NMMX),XGRDMM(NMMX),YGRDMM(NMMX),
     7  ZGRDMM(NMMX),HMM(LAYERS,NMMX),AVM(LAYERS,NMMX),
     8  AVN(LAYERS,NMMX),AVM1(LAYERS,NMMX),AVN1(LAYERS,NMMX),
     9  AVUE(LAYERS,NMMX),AVVE(LAYERS,NMMX),MMTIME,IBLK,
     T  NNTNUM,METDAT(20)
C
      INTEGER UPTIME, SFTIME, TSCSF, PWSF, VVSF, WWSF, GWTIME, BTSF,
     M  STUNUM
C
C     UPPER AIR VARIABLES
C
C     NUAX - NUMBER OF UA STATIONS SOUGHT
C     NUANUM - NUMBER OF SUITABLE UA STATIONS FOUND
C     STUNUM - ID NUMBER OF ITH STATION FOUND
C     XSTA(I) - UTM X COORDINATE OF UA STATION I  (KM)
C     YSTA(I) - UTM Y CORDINATE OF UA STATION (KM)
C     ZSTA(I) - ELEVATION (METERS)
C     UDIST(I) - DISTANCE TO SIMULATION SITE
C     UPTIME(I) - PACKED JULIAN DAY AND HOUR OF UA OBSERVATIONS (GREENWICH)
C     NTR(I) - NUMBER OF TEMPERATURE AND PRESSURE READINGS
C     IFLAGU(I) - FLAG DESCRIBING UA DATA
C     PRES(I,J) - UA PRESSURE AT LEVEL I FOR JTH STATION  (PASCALS)
C     TEMP(I,J) - UA TEMP (DEGREES K)
C     QHUM(I,J) - UA SPECIFIC HUMIDITY
C     ZMSL(I,J) - UA ELEVATION (METERS ABOVE SEA LEVEL)
C     VRTEMP(I,J) - UA VIRTUAL TEMP (DEGREES K)
C     POTEMP(I,J) - UA POTENTIAL TEMP (DEGREES K)
```

```
C     UWIND(I,J) - UA U-COMPONENT WIND AT ELEVAION ZMSL
C     VWIND(I,J) - UA V-COMPONENT WIND AT ELEVAION ZMSL
C
C     OBSERVATIONS AT SIGNIFICANT WIND LEVELS
C
C     NWR - NUMBER OF UA WIND READINGS
C     UVEL(I,J) - U-COMPONENT WIND AT LEVEL ZWDMSL
C     VVEL(I,J) - V-COMPONENT WIND AT LEVEL ZWDMSL
C     ZWDMSL(I,J) - ELEVATION OF SIGNIFICANT WIND READINGS
C
C     SURFACE VARIABLES
C
C     NSFX - NUMBER OF SURFACE STATIONS SOUGHT
C     NSFNUM - NUMBER OF SURFACE STATIONS FOUND
C     STSNUM(I) - ID NUMBER OF SURFACE STATION I
C     SDIST(I) - DISTANCE TO SIMULATION SITE
C     XSTS(I),YSTS(I) - UTM COORDINATES OF SURFACE STATIONS
C     ZSTS(I) - ELEVATION OF SURFACE STATION (METERS)
C     SFTIME(I) - TIME OF SURFACE STATION OBSERVATION (PACKED JULIAN DAY AND HOU
C     WDSF(I) - WIND DIRECTION (DEGREES FROM TRUE NORTH)
C     WSSF(I) - SPEED (MPS)
C     WGSF(I) - WIND GUSTS (MPS)
C     SLPSF(I) - SEA LEVEL PRESS (MB) (CONVERTED TO STATION PRESS. IN MB)
C     BTSF(I) - BAROMETRIC TENDENCY
C     TSSF(I) - SURFACE TEMP (DEG K)
C     TDSF(I) - DEW POINT DEPRESSION (DEG K)
C     ASSF(I) - ALTIMETER SETTING (INCHES HG)
C     PASF(I) - 6 HOURLY PRECIPITATION (MILLIMETERS)
C     TSCSF(I) - TOTAL SKY COVER
C     PWSF(I) - PAST WEATHER (WMO 4500)
C     VVSF(I) - VISIBILITY (M)
C     WWSF(I,J) - PRESENT WEATHER FOR STATION J (INTEGER CODE)
C     ICLOUD(I,J) - CLOUD DATA FOR STATION J (INTEGER CODE)
C         I=1:  FRACTION OF CELESTIAL DOME COVERED BY LOW CLOUDS (8 THS)
C         I=2:  TYPE OF LOW CLOUD (WMO 0513)
C         I=3:  HEIGHT ABOVE GROUND OF LOWEST CLOUD (WMO 1600)
C         I=4:  TYPE OF MIDDLE CLOUD (WMO 0515)
C         I=5:  TYPE OF HIGH CLOUD (WMO 0509)
C         I=6:  AMOUNT OF CLOUDS (8 THS)
C         I=7:  CLASSIFICATION OF CLOUDS ACCORDING TO HEIGHT
C         I=8:  TYPE OF CLOUD WITHIN HEIGHT CLASSIFICATION
C         I=9:  HEIGHT ABOVE GROUND OF BASE OF CLOUD LAYER (WMO 1677)
C
C     GLOBAL WEATHER VARIABLES
C
C     NGWX - NUMBER OF GWC VARIABLES SOUGHT
C     NGWNUM(I) - NUMBER OF GWC VARIABLES FOUND FOR TIME I
C     IGGNUM(I,J) - GRID POINT INDEX (INTEGER)
C     GDIST(I,J) - DISTANCE TO SIMULATION SITE (METERS)
C     XGRD(I,J),YGRD(I,J) - UTM COORDINATES OF GWC GRID POINT
```

```
C     ZGRD(I,J) -  SURFACE ELEVATION AT UTM XGRD,YGRD: SET EQUAL TO ZERO
C     IFLAGG(I,J) - FLAG FOR DATA FROM TIME I AND GRID POINT J
C                 - I=1 IS GWC DATA AT UPTIME
C                 - I=2 IS GWC DATA AT WINDEX SIMULATION TIME
C     GWTIME(I,J) - PACKED JULIAN DAY AND HOUR FOR TIME AND STATION
C     PGW(K,I,J) - PRESSURE AT INDICATED LEVEL, TIME, AND STATION
C     QGW(K,I,J) - SPECIFIC HUMIDITY
C     TGW(K,I,J) - TEMPERATURE (DEGREES K)
C     UGW(K,I,J),VGW(K,I,J) - U,V COMPONENTS OF WIND
C     ZZGW(K,I,J) - ELEVATION OF GWC PREDICTIONS (M)
C
C     MESOMODEL VARIABLES
C
C     NMMX - NUMBER OF MESOSCALE GRID POINTS SOUGHT
C     NMMNUM - NUMBER OF MESOSCALE GRID POINTS FOUND
C     RMDIST(J) - DISTANCE FROM GRID POINT J TO SIMULATION SITE (M)
C     XGRDMM(J),YGRDMM(J) - X,Y UTM COORDINATES OF MESO GRID POINT J
C     ZGRDMM(J) - SURFACE ELEVATION AT MESO GRID POINT
C     HMM(L,J) - LAYER HEIGHT (METERS)
C     AVM(L,J) - AVERAGE MOMENTUM FLUX, X DIRECTION (KG/MMS)
C     AVN(L,J) - AVERAGE MOMENTUM FLUX, Y DIRECTION (KG/MMS)
C     AVM1(L,J) - FIRST MOMENT MOMEMTUM FLUX, X DIRECTION (GRID)
C     AVN1(L,J) - FIRST MOMENT MOMEMTUM FLUX, Y DIPECTION (GRID)
C     AVUE(L,J) - HEIGHT INTEGRATED ENERGY FLUX, X DIRECTION (GRID) (J*M/S)
C     AVVE(L,J) - HEIGHT INTEGRATED ENERGY FLUX, Y DIRECTION (GRID) (J*M/S)
C     MMTIME(J) - PACKED JULIAN DAY AND HOUR OF MESOMODEL SIMULATION
C     IBLK - NUMBER OF BLOCKS STACKED
C
C     MESONET VARIABLES
C
C     NNTNUM - NUMBER OF MESNET DATA POINTS
C     METDAT(20) - DUMMY ARRAY TO CONTAIN MESNET DATA
 END
MRCHDR  PROC
      PARAMETER RUNS=25, NPARAM=6
      PARAMETER HDRSIZ=(10+NPARAM)*RUNS+1
      COMMON /MRCHDR/ RCOUNT,PTEMP(RUNS,3),ZELEV(RUNS,3),
     1                DELTAQ(RUNS),XWIND(RUNS),YWIND(RUNS),TERBLK(RUNS)
     2                ,PARAMS(RUNS,NPARAM)
      INTEGER RCOUNT
      INTEGER TERBLK
      INTEGER PARAMS
C
C     /MRCHDR/ CONTAINS DIRECTORY INFORMATION FOR THE MICRO WIND MODEL
C     OUTPUT FILE.  THE DIRECTORY RESIDES IN THE FIRST BLOCK OF THE
C     OUTPUT FILE: TEMPERATURE, U-WIND, AND V-WIND FIELDS ARE STORED IN
C     CONSECUTIVE BLOCKS THEREAFTER.  THE DIRECTORY CONTAINS THE DRIVING
C     PARAMETERS FOR EACH ARCHIVED RUN.
C
C     RCOUNT          - NUMBER OF RUNS ARCHIVED IN OUTPUT FILE
```

```
C     PTEMP(I,.)      -   POTENTIAL TEMPERATURE PROFILE
C     ZELEV(I,.)      -   FOR RUN I (DEGREES K: METERS)
C     DELTAQ(I)       - SURFACE HEATING FOR RUN I
C     XWIND(I)        - EAST WIND COMPONENT FOR RUN I (METERS/SEC)
C     YWIND(I)        - NORTH WIND COMPONENT FOR RUN I
C     TERBLK(I)       - TERRAIN BLOCK NUMBER FOR RUN I
C      PARAMS(I,1)          -    ILE FOR RUN I
C      PARAMS(I,2)          -    ILW FOR RUN I
C      PARAMS(I,3)          -    JLN FOR RUN I
C      PARAMS(I,4)          -    JLS FOR RUN I
C      PARAMS(I,5)          -    IHIERC FOR RUN I
C      PARAMS(I,6)          -    IRELAX FOR RUN I
C
 END
MRCIPF  PROC
      PARAMETER IMRCS=6
      COMMON /MRCIPF/ IDAT(IMRCS),IMDT(IMRCS),
     1  ILASTR,NTST,IFLUX,IHIERC,MTRBLK,IRELAX,ILE,ILW,JLN,JLS
C
C     /MRCIPF/ CONTAINS FLAGS WHICH CONTROL THE DATA BASE SEARCH AND
C     THE MICRO WIND SIMULATION
C
C     IDAT(1)         - UPPER AIR DATA ACCESS FLAG
C     IDAT(2)         - SURFACE STATION DATA ACCESS FLAG
C     IDAT(3)         - GWC DATA ACCESS FLAG
C     IDAT(4)         - MESOMODEL DATA ACCESS FLAG
C     IDAT(5)         - MESONET DATA ACCESS FLAG
C     IDAT(6)         - MANUAL OVERRIDE FLAG
C     IMDT(1)         - UA TIME INTERPOLATION FLAG
C     IMDT(2)         - SURFACE STATION TIME CRITERIA FLAG
C     IMDT(3)         - GWC FLAG (NOT USED)
C     IMDT(4)         - NUMBER OF BLOCK IN FILE GRDFL2 CONTAINING
C                       MESOMODEL DATA
C     IMDT(5)         - MESONET FLAG (NOT USED)
C     ILASTR          - MULTIPLE RUN FLAG
C     NTST            - ARCHIVING FLAG
C     IFLUX           - NUMBER OF LOCAL FLUX BOXES AVERAGED
C     IHIERC          - MODEL COMPLEXITY FLAG
C                       IF IHIERC = 0, WINDEX IS NOT EXECUTED
C     MTRBLK          - MICRO TERRAIN DATA ACQUISITION FLAG
C     IRELAX          - NUMBER OF RELAXATION PASSES TO BE EXECUTED IN
C                       WINDEX
C     ILE             - EASTERNMOST GRID INDEX LIMIT
C     JLN             - NORTHERNMOST GRID INDEX LIMIT
C     ILW             - WESTERNMOST GRID INDEX LIMIT
C     JLS             - SOUTHERNMOST GRID INDEX LIMIT
C
 END
OBUFER  PROC
C
```

```
C     UPPER AIR HEADER DATA IS LOADED INTO THIS BUFFER
C
C     UASTNS(I) - STATION NUMBER
C     UASTNA(I) - ANGLE BETWEEN TRUE N AND UTM N
C     UASTNX(I) - STATION UTM X COORD IN KM
C     UASTNY(I) - STATION UTM Y COORD IN KM
C     UASTNZ(I) - STATION ELEVATION IN METERS
C     TLVLMX(I,J) - NO. OF LEVELS OF TEMP. DATA FOR STATION, RUN
C     TADRES(I,J) - DRUM ADDRESS OF TEMP DATA FOR STATION, RUN
C     WLVLMX(I,J) - NO. OF LEVELS OF WIND DATA FOR STATION, RUN
C     WADRES(I,J) - DRUM ADDRESS OF WIND DATA FOR STATION, RUN
C     LATEST - RUN INDEX FOR LATEST RUN
C     RUNMAX - MAX. NO. OF RUNS
C     STNMAX - MAX. NO. OF STATIONS
C     UATIME(I) - PACKED JULIAN DAY AND HOUR OF UA OBSERVATIONS FOR RUN 1
C
      COMMON /OBUFER/ UASTNS(SU),UASTNA(SU),UASTNX(SU),UASTNY(SU),
     1  UASTNZ(SU),TLVLMX(SU,RU),TADRES(SU,RU),WLVLMX(SU,RU),
     2  WADRES(SU,RU),UATIME(RU),LATEST,RUNMAX,STNMAX
      INTEGER TLVLMX,WLVLMX,TADRES,WADRES,LATEST,RUNMAX,STNMAX,UATIME,
     1  UASTNS
 END
SBUFER  PROC
      COMMON /SBUFER/ SFCSTN(SS),SFSTNA(SS),SFSTNX(SS),SFSTNY(SS),
     1  SFSTNZ(SS),LATEST,RUNMAX,STNMAX
      INTEGER SFCSTN,LATEST,RUNMAX,STNMAX
C
C     SFCSTN(I) - SURFACE STATION ID
C     SFSTNA(I) - ANGLE BETWEEN TRUE N AND UTM N
C     SFSTNX(I) - UTM X
C     SFSTNY(I) - UTM Y
C     SFSTNZ(I) - STATION ELEVATION (METERS)
C     RUNTIM(I) - PACKED JULIAN DAY AND HOUR OF SS OBSERVATION FOR RUN I
C     LATEST - LATEST RUN CONTAINING DATA
C     RUNMAX - NUMBER OF RUNS IN DATA BASE
C     STNMAX - NUMBER OF STATIONS IN DATA BASE
C     SADRES(I,J) - ARRAY OF POINTERS FOR STATION I, RUN J
C
 END
SCRTCH  PROC
        COMMON/SCRTCH/IDXSTA(SS),STADST(SS)
C
C       SCRATCH  STORAGE AREA USED IN PARTICULAR
C        BY SUBROUTINE CLSSTN
C
 END
SFDATA  PROC
C
C     SURFACE DATA
C
```

```
C     STIME - TIME
C     WD - WIND DIRECTION (DEG)
C     WS - SPEED (MPS)
C     WG - WIND GUSTS (MPS)
C     SLP - SEA LEVEL PRESS (MB) (CONVERTED TO STATION PRESS. IN MB)
C     BT - BAROMETRIC TENDENCY
C     TS - SURFACE TEMP (DEG K)
C     TD - DEW POINT DEPRESSION (DEG K)
C     AS - ALTIMETER SETTING (INCHES HG)
C     PA - 6 HOURLY PRECIP. (MILLIMETERS)
C     TSC - TOTAL SKY COVER
C     PW - PAST WEATHER (WMO 4500)
C     VV - VISIBILITY (M)
C     WW(I)  - PRESENT WEATHER
C     NH - FRACTION OF CELESTIAL DOME COVERED BY LOW CLOUDS (8 THS)
C     CL - TYPE OF LOW CLOUD (WMO 0513)
C     H - HEIGHT ABOVE GROUND OF LOWEST CLOUD (WMO 1600)
C     CM - TYPE OF MIDDLE CLOUD (WMO 0515)
C     CH - TYPE OF HIGH CLOUD (WMO 0509)
C     NS - AMOUNT OF CLOUDS (8 THS)
C     K - CLASSIFICATION OF CLOUDS ACCORDING TO HEIGHT
C     CT - TYPE OF CLOUD WITHIN HEIGHT CLASSIFICATION
C     HSHS - HEIGHT ABOVE GROUND OF BASE OF CLOUD LAYER (WMO 1677)
C
      COMMON /SFDATA/ STIME,WD,WS,WG,SLP,BT,TS,TD,AS,PA,TSC,PW,VV,WW(4)
     1  NH,CL,H,CM,CH,NS,K,CT,HSHS
      INTEGER STIME,TSC,PW,VV,WW,CL,H,CM,CH,CT,HSHS,BT
 END
TBLHDR  PROC
      PARAMETER IDIM=50, THDRSZ=251
      COMMON /TBLHDR/ NNBLK,NDIMXH(IDIM),NDIMYH(IDIM),DELH(IDIM),
     1               SWUTMX(IDIM),SWUTMY(IDIM)
C
C      NNBLK              - NUMBER OF ARCHIVED TERRAIN AND ROUGNESS PAIRS
C      NDIMXH(.)          - X DIMENSION OF TERRAIN ARRAYS
C      NDIMYH(.)           - Y DIMENSION OF TERRAIN ARRAYS
C      DELH(.)             - GRID SPACING (METERS) OF TERRAIN DATA
C     SWUTMX(.)          - UTM X COORDINATE OF SW CORNER
C     SWUTMY(.)          - UTM Y COORDINATE OF SW CORNER
C
 END
TDATA   PROC
C**   UPPER AIR DATA FOR PARTICULAR STATION, RUN
C     PS - PRESSURE IN PASCALS (PASCALS*.01 = MB)
C     QS - SPECIFIC HUMIDITY
C     TS - TEMP DEG K
C     ZS - ELEVATION METERS
C     TV - VIRTUAL TEMP DEG K
C     TA - POTENTIAL TEMP. DEG. K
C     US,VS - WIND COMPONENTS MPS
```

```
C     DELTA - HYDROSTATIC EQUATION EXPONENT
C     PD - D-VALUE FOR PRESSURE LEVEL
C     IMAX - NO. OF LEVELS
C
        COMMON/TDATA/ TTIME,
     1  PS(40),QS(40),TS(40),ZS(40),TV(40),TA(40),US(40),
     2  VS(40),DELTA(40),PD(40),IMAX
        INTEGER TTIME
 END
TERDAT  PROC
C
      DIMENSION H(ILL,JLL)
      COMMON /TERDAT/ TMATRX(ILL,JLL), ROUGH(ILL,JLL)
      EQUIVALENCE (H,TMATRX)
C
C     H(I,J)          - TERRAIN HEIGHT AT GRID POINT (I,J) (METERS)
C     ROUGH(I,J)      - ROUGHNESS AT GRID POINT (I,J)
C
 END
TRPDTA  PROC
      COMMON /TRPDTA/ PT(3),QT(3),TT(3),ZT(3),TB(3),TI(3),UT(3),VT(3),
     1  TD(3),NMAX
 END
WDATA  PROC
C     UPPER AIR DATA   SIGNIFICANT WIND READINGS
C     UW,VW - WIND COMPONENTS MPS
C     ZW - ELEVATION METERS
C     KMAX - NO. OF LEVELS
C
      COMMON /WDATA/ WTIME,UW(40),VW(40),ZW(40),KMAX
 END
```

```
DATPRM  PROC
C
C      THIS PROCEDURE SETS PARAMETERS RELATED TO THE DATA BASE
C      STRUCTURE AND SEARCH.
       PARAMETER SS=168.SOBS=5*SS+3
       PARAMETER SU=25. RU=3. UOBS=(4*RU+5)*SU+RU+3
       PARAMETER PG=25. RG=80.GLVLS=4.GPTS=2*(PG+RG+1).GBLK=5700
       PARAMETER MXBLKU=30. MXBLKS=1. MXBLKG=1
       PARAMETER TIMDIF=7
C
C      MXBLKU - MAXIMUM NO. OF DATA BLOCKS IN FILE OBFILE
C      MXBLKG - MAXIMUM NO. OF DATA LOCKS IN FILE GWCPFL
C      MXBLKS - MAXIMUM NO. OF DATA BLOCKS IN FILE SFCOBF
C
 END
GRDPRM  PROC
C      THIS PROCEDURE SETS PARAMETERS GIVING DIMENSIONS FOR THE MICRO
C      TERRAIN AND WIND FIELD ARRAYS.
C
       PARAMETER ILL=40. JLL=40. ICC=ILL-1. JCC=JLL-1.
     1          NFOUT = ILL*JLL. ICCJCC=ICC*JCC
 END
UAOBPR PROC
       PARAMETER SU=25. RU=3. NU = 66 . LYR2=59*NU+2
       PARAMETER UOBS=(4*RU+5)*SU+RU+3.LYR4=20*NU+70
 END
GRIDPR PROC
       PARAMETER IPR=25. JPR=25. NGRD=IPR*JPR. NGP1=NGRD+1
       PARAMETER NGMA=NGRD+IPR+3.NJL=NGRD-IPR NJLP1=NJL+1.NLB=NGMA-2
       PARAMETER IPP1=IPR+1.NRL=NGRD+IPP1.IPP2=IPP1+1.IPM1=IPR-1
 END
GWCPPR PROC
       PARAMETER PG=25. RG=80.GLVLS=4.GPTS=2*(PG+PG+1).GBLK=5700
 END
SFOBPR PROC
       PARAMETER SS=168. RS=4.SOBS=5*SS+3.SBLK=6800
       PARAMETER DBAST1 = 30500.DBAST2 = 33500.DBAST3 = 37000
 END
ROTATE PROC
       REAL ROTATE/0.0/
 END
```

*This procedure element is used when the analysis is run in the soutnwestern United States

```
DATPRM  PROC
C
      PARAMETER SS=387,SOBS=5*SS+3
      PARAMETER SU=36, RU=3, UOBS=(4*RU+5)*SU+RU+3
      PARAMETER PG=25, RG=80,GLVLS=4,GPTS=2*(PG+RG+1),GBLK=5700
      PARAMETER MXBLKU=30, MXBLKS=1, MXBLKG=1
      PARAMETER TIMDIF=-1
C
C     MXBLKU - MAXIMUM NO. OF DATA BLOCKS IN FILE OBFILE
C     MXBLKG - MAXIMUM NO. OF DATA LOCKS IN FILE GWCPFL
C     MXBLKS - MAXIMUM NO. OF DATA BLOCKS IN FILE SFCOBF
C
 END
GRDPRM  PROC
C
      PARAMETER ILL=51, JLL=31, ICC=ILL-1, JCC=JLL-1
     1          ,NFOUT = ILL*JLL, ICCJCC=ICC*JCC
 END
UAOBPR PROC
      PARAMETER SU=36, RU=3, NU = 72 , LYR2=59*NU+2
      PARAMETER UOBS=(4*RU+5)*SU+RU+3,LYR4=20*NU+70
 END
GRIDPR PROC
      PARAMETER IPR=25, JPR=25, NGRD=IPR*JPR, NGP1=NGRD+1
      PARAMETER NGMA=NGRD+IPR+3,NJL=NGRD-IPR,NJLP1=NJL+1,NLB=NGMA-2
      PARAMETER IPP1=IPR+1,NRL=NGRD+IPP1,IPP2=IPP1+1,IPM1=IPR-1
 END
GWCPPR PROC
      PARAMETER PG=25, RG=80,GLVLS=4,GPTS=2*(PG+RG+1),GBLK=5700
 END
SFOBPR PROC
      PARAMETER SS=387, RS=1,SOBS=5*SS+3,SBLK=6600
      PARAMETER DBAST1 = 30500,DBAST2 = 32000,DBAST3 = 33500
 END
ROTATE PROC
      REAL ROTATE/.89982/
 END
```

***This procedure element is used when the analysis is run in Europe.**

```
ACNRML  PROC
C**   GLOBAL ARRAY OF NORMAL ACCELERATIONS
      COMMON /ACNRML/ ACNRML(ICC,JCC)
 END
INPPRM  PROC
      COMMON /INPPRM/ NRELAX,DGRID,ZGRID,DR,DD,ILEVEL,
     1      IL,JL,IC,JC,IC1,JC1,NR,TRLX,VRLX
C
C     /INPPRM/ CONTAINS PARAMETERS LOCAL TO WINDEX AND ITS SUBROUTINES
C
C          IL          -    'DO LOOP' PARAMETERS IN SUBROUTINE RELAX
C          JL          -    USUAL VALUES ARE ILL,JLL
C          IC,IC1      - SET EQUAL TO ILE-1,ILW
C          JC,JC1      - SET EQUAL TO JLN-1,JLS
C          NRELAX      - NUMBER OF RELAXATION PASSES TO BE EXECUTED
C          NR          - CURRENT RELAXATION PASS COUNTER
C          DGRID       - MICRO TERRAIN GRID SPACING
C          DD          - ROTATED GRID SPACING
C                        DD = DGRID*SQRT(2)
C          DR          - SET EQUAL TO 1/DD
C          ZGRID       - SURFACE LAYER HEIGHT; SET EQUAL TO 6 METERS
C          ILEVEL      - MODEL COMPLEXITY FLAG
C                        ILEVEL = 1:  CONSTANT TEMPERATURE FIELD
C                        ILEVEL = 2:  TEMPERATURE FIELD SUBJECT TO
C                       VARIATIONAL MINIMIZATION
C          TRLX,VRLX   - RELAXATION PARAMETERS
C
 END
SETBOX  PROC
      COMMON /SETBOX/ NSET(4),IWTFLG,RWTFLG,BOXAV
C
C       NSET(.)              -   FLAGS SPECIFYING WHICH LOCAL FLUX BOXES TO
C                                USE IN ESTIMATING LOCAL RESIDUALS
C                            = 1 USE
C                            = 0 DON'T USE
C       IWTFLG               -   FLAG USED TO CONTROL SELECTION OF LOCAL
C                                FLUX BOXES
C                            = 2 CHOOSE TWO UPSTREAM BOXES
C                            = 4 CHOOSE ALL FOUR LOCAL BOXES
C       RWTFLG               -   SET EQUAL TO IWTFLG
C       BOXAV                -   FACTOR USED IN AVERAGING LOCAL RESIDUALS.
C                                SET IN ROUTINE INPUTS
C
 END
START   PROC
C**   MESOSCALE DRIVERS USED BY SETUP TO INITIALIZE WIND AND
C**   TEMPERATURE FIELDS
      COMMON /START/ VE,VN,DELTQ
 END
DRIVER  PROC
      COMMON /DRIVER/ UB,VB,VMAG,VMAGSQ
```

```
C     /DRIVER/ CONTAINS VARIABLES DESCRIBING THE BACKGROUND
C     DRIVING WIND
C
C     VMAG            - MAGNITUDE OF DRIVING WIND
C     VMAGSQ          - EQUALS VMAG**2
C     UB              - U COMPONENT OF DRIVING WIND (ROTATED SYSTEM)
C     VB              - V COMPONENT OF DRIVING WIND (ROTATED SYSTEM)
C
 END
SLOPES  PROC
C**   GLOBAL ARRAY OF TERRAIN SLOPES AND WARPED COORDINATE AREA FACTORS
      COMMON /SLOPES/ SLOPE1(ICC,JCC), SLOPE2(ICC,JCC),
     1                AREAF(ICC,JCC)
 END
TEMPEX  PROC
      COMMON /TEMPEX/ TEMPEX(ICC,JCC)
 END
FIELDS   PROC
      COMMON /WNDVEL/ U(ICC,JCC), V(ICC,JCC)
      COMMON /POTEMP/ THTA(ICC,JCC)
C
C     U(...)          - ARRAY OF UWIND COMPONENTS (ROTATED SYSTEM)
C     V(...)          - ARRAY OF VWIND COMPONENTS (ROTATED SYSTEM)
C     THTA(...)       - SURFACE POTENTIAL TEMPERATURE ARRAY
C
 END
SURFOB  PROC
      PARAMETER NOBS=10
      COMMON /SURFOB/ KINDX(NOBS),LINDX(NOBS),UW(NOBS)
     1     ,VW(NOBS),THETA0(NOBS),NUMOBS
C
C     /SURFOB/ VARIABLES ARE SUPPLIED DATA VIA MANUAL INPUT
C     AND CAUSE PEGGING OF WIND AND TEMPERATURE VALUES
C
C     KINDX(I)         - THE ITH PEGGED VALUE IS AT GRID
C     LINDX(I)         - POINT (KINDX(I),LINDX(I))
C     UW(I)            - U COMPONENT OF ITH PEGGED WIND VECTOR
C     VW(I)            - V COMPONENT OF ITH PEGGED WIND VECTOR
C     THTA(I)          - POTENTIAL TEMPERATURE OF ITH PEGGED POINT
C     NUMOBS           - NUMBER OF PEGGED OBSERVATIONS
 END
DERIV   PROC
      COMMON /DERIV/ DU(ICC,JCC),DV(ICC,JCC),DT(ICC,JCC)
C
C     DU(I,J)          - PARTIAL DERIVATIVE OF ACCELERATION RESIDUAL
C                        WITH RESPECT TO TO U(I,J)
C     DV(I,J)          - PARTIAL DERIVATIVE OF ACCELERATION RESIDUAL
C                        WITH RESPECT TO TO V(I,J)
C     DT(I,J)          - PARTIAL DERIVATIVE OF TEMPERATURE RESIDUAL
C                        WITH RESPECT TO TO THTA(I,J)
```

```
C
 END
MNSRCH  PROC
      COMMON /MNSRCH/ RES(100),ENERGY(100),VDRSUM(100),
     1                TRES(100)
C     GLOBAL RESIDUES FOR EACH RELAXATION STEP
C     RES(.)              -   ACCELERATION RESIDUE
C     TRES(.)             -   TEMPERATURE RESIDUE
C     VDRSUM(.)           -   SUM SQUARE OF DERIVATIVES
C     ENERGY(.)           -   SUM SQUARE OF SURFACE PARALLEL VELOCITY
C                             MAGNITUDES
 END
LOCAL   PROC
      COMMON /LOCAL/ HD1(9),HD2(9),AR(9),EN(9),ER(9),
     1     THD(9),BUOY(9),UL(9),VL(9),TH(9)
C
C**   LOCAL QUANTITIES
C
C     HD1(.)              -    U-SLOPE OF TERRAIN
C     HD2(.)              -    V-SLOPE OF TERRAIN
C     AR(.)               -    AREA FACTOR DUE TO WARPED GEOMETRY
C     EN(.)               -    PROFILE EXPONENT
C     ER(.)               -    SQRT(4*EN+1)
C     THD(.)              -    TEMPERATURE DERIVATIVE FROM PROFILE
C     BUOY(.)             -    BUOYANCY
C     UL(.)               -    LOCAL U-WIND
C     VL(.)               -     LOCAL V-WIND
C     TH(.)               -    LOCAL TEMPERATURE
 END
SAVEM   PROC
C**   SAVED FIELDS OCCUR AT RELAXATION STEP NSAVE
      COMMON /SAVEM/ USAVE(ILL,JLL),VSAVE(ILL,JLL),
     1               TPSAVE(ILL,JLL),NSAVE
 END
```

D.3 SUBROUTINES OF MRC*EPAMS

D.3.1 Executive and Data Access Routines (XECAMS, ANALYZ, WNDMGR, TERACQ, DTBMGR)

```
      SUBROUTINE ANALYZ
C
      INCLUDE MESAGE
C
      GO TO (20.30.40). TASK
C
 20   CALL MIXLYR
      RETURN
C
 30   CALL WNDMGR
      RETURN
C
 40   CONTINUE
      RETURN
      END
```

```
        SUBROUTINE APPLY
        RETURN
        END
      SUBROUTINE CLSSTN(NSTATN,X,Y,IDXSTA,STADST,STNMAX,DISTMN)
C
C*****************************************************************************
C*
C*    CLSSTN IDENTIFIES THOSE POINTS FROM A GIVEN LIST WHICH LIE
C*    INSIDE A PRESCRIBED CIRCLE.
C*
C*    INPUTS:
C*              X(.) = X COORDINATES OF INPUT POINTS
C*              Y(.) = Y COORDINATES OF INPUT POINTS
C*           DSTMIN = RADIUS OF SPECIFIED CIRCLE
C*           XPLACE =   CENTER COORDINATES
C*           YPLACE =   OF SPECIFIED CIRCLE
C*           STNMAX = NUMBER OF POINTS IN INPUT LIST
C*
C*    OUTPUTS:
C*           NSTATN = NUMBER OF POINTS LYING INSIDE SPECIFIED CIRCLE
C*        IDXSTA(.) = INDICES OF POINTS LYING INSIDE CIRCLE,
C*                    ARRANGED IN ORDER OF INCREASING DISTANCE FROM
C*                    CENTER; CLOSEST POINT IS X(IDXSTA(1)),Y(IDXSTA(1))
C*        STADST(I) = DISTANCE BETWEEN CENTER AND ITH CLOSEST POINT
C*
C*****************************************************************************
C
      INCLUDE DATPRM
      INCLUDE COORD
C
C
      DIMENSION X(1),Y(1), IDXSTA(1),STADST(1)
C
      INTEGER STNMAX
C
C**   DETERMINE THE CLOSEST STATION WITHIN DISTANCE DISTMN AND ORDER
C**   STATIONS IN ASCENDING ORDER OF DISTANCE
C
      NSTATN = 0
      XSOURS = XPLACE
      YSOURS = YPLACE
      N = STNMAX
      DO 100 I=1,N
C**   CALCULATE DISTANCE FROM RELEASE TO STATION
      D = SQRT((X(I)-XSOURS)**2+(Y(I)-YSOURS)**2)
      IF (D .GT. DISTMN) GO TO 100
      NSTATN = NSTATN +1
      IDXSTA(NSTATN) = I
      STADST(NSTATN) = D
  100 CONTINUE
      IF (NSTATN .LT. 2 ) GO TO 120
```

```
      DO 110 J=2.NSTATN
      DO 110 I=2.NSTATN
      IF (STADST(I-1) .LE. STADST(I)) GO TO 110
      D = STADST(I)
      K = IDXSTA(I)
      STADST(I) = STADST(I-1)
      IDXSTA(I) = IDXSTA(I-1)
      STADST(I-1) = D
      IDXSTA(I-1) = K
  110 CONTINUE
  120 RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE DRIVRS
C
C*****************************************************************************
C*
C*    SUBROUTINE DRIVRS COMPARES THE DRIVING PARAMETERS GENWND, DELQ,
C*    T, Z, MTRBLK DETERMINED BY DATANL FOR THIS RUN WITH THE DRIVING
C*    PARAMETERS OF PREVIOUSLY ARCHIVED RUNS.  IF THE DRIVERS FOR A
C*    PREVIOUS RUN AGREE SUFFICIENTLY WITH THE NEW DRIVERS, EXECUTION
C*    OF WINDEX IS SUPPRESSED.
C*
C*****************************************************************************
C
      INCLUDE MRCHDR
      INCLUDE INTLYZ
      INCLUDE MRCIPF
      INCLUDE FILES
      DATA CONST1/.2/, CONST2/.1745/, CONST3/1.5/
C
C
C**   DETERMINE WHETHER NEW DRIVING PARAMETERS MATCH AN
C**   ARCHIVED SET
      IF(RCOUNT.EQ.0) GO TO 200
      IF(NTST.EQ.1) GO TO 200
      VMAG1 = GENWND(1)**2 + GENWND(2)**2
      ANGLE1 = ATAN2(GENWND(1),GENWND(2))
      DO 100 I=1,RCOUNT
C**   COMPARE TERRAIN.  IF DIFFERENT, KEEP SEARCHING
      IF(MTRBLK.NE.TERBLK(I)) GO TO 100
C**   COMPARE BACKGROUND WIND MAGNITUDES
      VMAG2 = XWIND(I)**2 + YWIND(I)**2
      VDIFF = ABS(VMAG1-VMAG2)
C**   IF MAGNITUDES DIFFER BY 20 PER CENT OR MORE, KEEP SEARCHING
      IF(VDIFF/VMAG1.GT.CONST1) GO TO 100
C**   COMPARE BACKGROUND WIND DIRECTIONS
      ANGLE2 = ATAN2(XWIND(I),YWIND(I))
      ANGDIF = ABS(ANGLE1-ANGLE2)
C**   IF DIRECTIONS DIFFER BY MORE THAN 10 DEGREES (.1745 RADIANS),
C**   KEEP SEARCHING
      IF(ANGDIF.GT.CONST2) GO TO 100
C**   COMPARE BUOYANCY DRIVERS
      DLQDIF = ABS(DELQ-DELTAQ(I))
      DLTDIF = DLQDIF*1.06
C**   IF BUOYANT TEMPERATURE DIFFERENCE EXCEEDS 1.5 DEGREES,
C**   KEEP SEARCHING
      IF(DLTDIF.GT.CONST3) GO TO 100
C
C**   ARCHIVED DRIVERS AGREE WITH NEW DRIVERS:
C**   WRITE OUT MATCHING BLOCK AND SUPPRESS EXECUTION OF WINDEX
      ISAME = I
      WRITE(XOTFIL,1000) ISAME
      IHIERC = 0
      RETURN
  100 CONTINUE
```

```
C
C
C
  200 CONTINUE
C**   NONE OF ARCHIVED DRIVERS MATCH:  UPDATE DIRECTORY WITH NEW DRIVERS
      RCOUNT = RCOUNT+1
      IF(RCOUNT.GT.25) RCOUNT=25
      I = RCOUNT
      XWIND(I) = GENWND(1)
      YWIND(I) = GENWND(2)
      DELTAQ(I) = DELQ
      TERBLK(I) = MTRBLK
      DO 150 J=1,3
      PTEMP(I,J) = TPRO(J)
      ZELEV(I,J) = ZPRO(J)
  150 CONTINUE
      PARAMS(I,1) = ILE
      PARAMS(I,2) = ILW
      PARAMS(I,3) = JLN
      PARAMS(I,4) = JLS
      PARAMS(I,5) = IHIERC
      PARAMS(I,6) = IRELAX
C
C
      WRITE(XOTFIL,2000) RCOUNT
      RETURN
C
 1000 FORMAT(//' DRIVING PARAMETERS AGREE WITH THOSE FOR ,
     1          ' ARCHIVED RUN NUMBER ',I2//)
 2000 FORMAT(//' DRIVERS ARE DIFFERENT THAN THOSE OF PREVIOUSLY '
     1          , 'ARCHIVED RUNS'/ ' ARCHIVE NUMBER FOR THIS RUN = ',I3)
C
      END
```

```
      SUBROUTINE DTBMGR
C
      INCLUDE DATPRM
      INCLUDE MESAGE
      INCLUDE COORD
      INCLUDE MRCIPF
      INCLUDE MRCDAT
      INCLUDE FILES
C
      PARAMETER DATBLK=1820
      DIMENSION IYIYI(DATBLK)
      EQUIVALENCE (IYIYI(1),NUANUM)
C
C**   INITIALIZE:
      DO 10 I=1,DATBLK
      IYIYI(I) = 0
   10 CONTINUE
C
      IF(IDAT(1).EQ.0) CALL MRCUA
      IF(IDAT(2).EQ.0) CALL MRCSFC
      IF(IDAT(3).EQ.0) CALL MRCGWC
      IF(IDAT(4).EQ.0) CALL MESMOD
      IF(IDAT(5).EQ.0) CALL MRCNET
C
C
C**   DATA SEARCH COMPLETE
C**   CALL DATA ANALYSIS ROUTINE
      CALL DATANL
C
C
      RETURN
      END
```

```
      SUBROUTINE HEC
      RETURN
      END
    COMPILER (DIAG=3)
    SUBROUTINE GWCDTA(GRID,RUN,FLAG)
    INCLUDE GWCPPR
    INCLUDE FILES
    INCLUDE GBUFER
    INCLUDE GWDATA
    INTEGER FLAG,GRID,RUN,ZS(4)
    INTEGER A(5700),B(4,6),X,Y
    INTEGER ADRES,BLOCK
    INTEGER OCT/O777777000000/
    DATA P /100000..85000..70000..50000./
    DATA ZS/110,1460,3010,5570/
    IF(RUN.EQ.BLOCK) GO TO 50
    BLOCK = RUN
    CALL BLKIN(5700,A(1),BLOCK,GWCFIL,I)
 50 CONTINUE
    ADRES = GTIME - ANLTIM(RUN)
    IF(ADRES.LT.0) GO TO 800
    IF(ADRES.GT.24) ADRES = ADRES - 76
    IF(ADRES.GT.18) GO TO 800
    ADRES = 300*ADRES + 12*(GRID-1)
    DO 150 J = 1,6
    DO 100 K = 1,2
    L = K+K
    M = 2*(J-1) + ADRES
    X = FLD(18,18,A(M+K))
    Y = FLD(0,18,A(M+K))
    IF(X.GT.32768) X = OR(X,OCT)
    IF(Y.GT.32768) Y = OR(Y,OCT)
    B(L-1,J) = X
    B(L,J) = Y
100 CONTINUE
150 CONTINUE
    DO 200 L = 1,4
    U(L) = B(L,1)
    V(L) = B(L,2)
    Z(L) = B(L,4) + ZS(L)
    T(L) = B(L,5)
    Q(L) = B(L,6)
    Q(L) = T(L) - Q(L)
    IF(Q(L).LT.0.0) Q(L) = 0.0
    IF(Q(L).GT.0.0.AND.P(L).GT.0.0) Q(L) = 62.2*SATVP(Q(L))/P(L)
    Q(L) = Q(L)/(1. - 0.61*Q(L))
200 CONTINUE
    FLAG = 1
    IF(RUN.EQ.41) FLAG = 0
    RETURN
800 CONTINUE
    WRITE(6,900)
    STOP
900 FORMAT( 'IMPROPER GWC TIME OR BLOCK')
    END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE SFCDTA(TIME,STN,FLAG)
      INCLUDE SFOBPR
      INCLUDE FILES
C   THE PURPOSE OF THIS ROUTINE IS TO EXTRACT SURFACE OBSERVATION DATA
C**   FROM THE DATA BASE
      INCLUDE SFDATA
      INCLUDE SBUFER.LIST
      INTEGER A(SBLK)
      INTEGER JUNK/-1/,RUN,FLAG,STN
      INTEGER  BLOCK,DAY,HOUR,NEWBLK,TIME
      INTEGER ADRES,PRVSTN,PRVTIM
      DATA BLOCK /99999/
C  FIND THE ADDRESS FOR THE DATA CORRESPONDING TO THE GIVEN STATION AND
C   TIME . IF THIS ADDRESS HAS NOT BEEN SET (IT IS ZERO) , RETURN TO
C   THE CALLING PROGRAM . OTHERWISE , FIND J , WHERE THE FIRST WORD OF
C   THE 11-WORD LOGICAL RECORD IS THE J-TH ELEMENT IN ARRAY A .
      IF(TIME.NE.PRVTIM) GO TO 2
      IF(STN.NE.PRVSTN) GO TO 10
      J = ADRES
      IF(J.NE.0) GO TO 12
      PRVSTN = 0
      FLAG = 0
      RETURN
    2 CONTINUE
      PRVTIM = TIME
      HOUR = TIME - DBAST3
      IF(HOUR.GE.0) GO TO 5
      HOUR = TIME - DBAST2
      IF(HOUR.GE.0) GO TO 5
      HOUR = TIME - DBAST1
    5 CONTINUE
      DAY = HOUR/100
      HOUR = HOUR - 76*DAY
      NEWBLK = HOUR/RS
      RUN = HOUR - RS*NEWBLK
      IF(NEWBLK.EQ.BLOCK) GO TO 10
      BLOCK = NEWBLK
      CALL BLKIN(SBLK,A,BLOCK+2,SFCFIL,I)
   10 CONTINUE
      J = 11*(RUN*SS + STN - 1) + 1
   12 CONTINUE
      ADRES = FLD(9,15,A(J))
      IF(ADRES.EQ.0) GO TO 14
      PRVSTN = STN
      FLAG = FLD(0,9,A(J))
      GO TO 30
   14 CONTINUE
      PRVSTN = 0
      FLAG = FLD(5,4,A(J))
```

```
   30 CONTINUE
      STIME = FLD(24,6,A(J+1))
    1 FORMAT(2I6)
C**   ETURN TO THE CALLING PROGRAM IF THE VALUE OF FLAG IS ZERO
      IF (FLAG.EQ.0)   RETURN
C  EXTRACT THE DATA IN THE (SURFACE) MANDATORY WORD 1
      CALL TESTFD(0,12,A(J+2),INDI)
      WD = INDI
      CALL TESTFD(12,12,A(J+2),INDI)
      IF (INDI.EQ.JUNK) INDI = -10
      WS = FLOAT(INDI)/10.
      CALL TESTFD(24,12,A(J+2),INDI)
      IF (INDI.EQ.0) INDI = -10
      WG = FLOAT(INDI)/10.
C   EXTRACT THE DATA IN THE (SURFACE) MANDATORY WORD 2
      CALL TESTFD(0,18,A(J+3),INDI)
      IF (INDI.EQ.JUNK) INDI = -10
      SLP = 10*INDI
      CALL TESTFD(18,18,A(J+3),BT)
      IF (BT.EQ.0) BT = JUNK
C   EXTRACT THE DATA IN THE (SURFACE) MANDATORY WORD 3
      CALL TESTFD(0,18,A(J+4),INDI)
      IF (INDI.EQ.JUNK) INDI = -10
      TS = FLOAT(INDI)/10.
      CALL TESTFD(18,18,A(J+4),INDI)
      IF (INDI.EQ. JUNK) INDI = -10
      TD = FLOAT(INDI)/10.
C   EXTRACT THE DATA IN THE (SURFACE) MANDATORY WORD 4
      CALL TESTFD(0,18,A(J+5),INDI)
      IF (INDI.EQ.JUNK) INDI = -100
      AS = FLOAT(INDI)/100.
      CALL TESTFD(18,18,A(J+5),INDI)
      PA = INDI
C**   XTRACT THE DATA IN THE (SURFACE) MANDATORY WORD 5
      CALL TESTFD(0,6,A(J+6),TSC)
      CALL TESTFD(6,6,A(J+6),PW)
      CALL TESTFD(18,18,A(J+6),VV)
C  EXTRACT THE DATA IN THE (SURFACE) MANDATORY WORD 6   TEST THAT THERE
C**   RE CODED DATA ON THE PRESENT WEATHER
      IF (A(J+7).NE.0) GO TO 15
      DO 20 I = 1,4
   20 WW(I) = JUNK
         GO TO 16
   15 DO 25 I = 1,4
      CALL TESTFD(IABS(I-1)*9,9,A(J+7),WW(I))
   25 CONTINUE
C  EXTRACT THE DATA IN THE (SURFACE) MANDATORY WORD 8
   16 CALL TESTFD(6,6,A(J+9),NH)
      CALL TESTFD(12,6,A(J+9),CL)
```

```
      CALL TESTFD(18.6,A(J+9),H)
      CALL TESTFD(24.6,A(J+9),CM)
      CALL TESTFD(30.6,A(J+9),CH)
C  EXTRACT THE DATA IN THE (SURFACE) MANDATORY WORD 9
      CALL TESTFD(6.6,A(J+10),NS)
      CALL TESTFD(12.6,A(J+10),K)
      CALL TESTFD(18.6,A(J+10),CT)
      CALL TESTFD(24.12,A(J+10),HSHS)
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE UADATA(STN,RUN,FLAG)
      INTEGER  A(5000)
      INCLUDE UAOBPR
      INCLUDE OBUFER
      INCLUDE FILES
      INCLUDE TDATA
      INCLUDE WDATA
      INCLUDE TRPDTA
      REAL     DP(40),D(40),S(40)
      REAL     ANGLE,OFFSET,PO,RADPDG
      REAL     GAMA,PSTD(2),TSTD(2),ZSTD
      INTEGER  BLOCK,NEWBLK
      INTEGER  STN,RUN
      INTEGER TADS,WADS
      INTEGER  FLAG
      EQUIVALENCE (DP,OS),(D,UW),(S,VW)
      DATA PO,RADPDG /100000.,0.01745329252/
      DATA GAMA, PSTD,           TSTD,          ZSTD
     *   /.0065, 101325.,20980., 288.15,216.65, 11476.
      DATA CONSTA /0.0341672473/
C  EXTRACT DATA
      NEWBLK = TADRES(STN,RUN)/10000
      IF(NEWBLK.EQ.BLOCK) GO TO 50
      BLOCK = NEWBLK
      CALL BLKIN(5000,A(1),BLOCK,UAFILE,ISTAT)
   50 CONTINUE
      NEWBLK = NEWBLK*10000
      TADS = TADRES(STN,RUN) + 1 - NEWBLK
      WADS = WADRES(STN,RUN) + 1 - NEWBLK
      FLAG = FLD(2,2,A(TADS-1))
      IF(FLD(0,6,A(WADS)).EQ.0) GO TO 60
      D(1) = 0.0
      S(1) = 0.0
      ZW(1) = UASTNZ(STN)
      KMAX = KMAX +  1
      GO TO 70
   60 CONTINUE
      FLAG = FLAG+4
   70 CONTINUE
      IF(IMAX.GT.40) IMAX = 40
      CALL XTRCUA(A(TADS),PS,IMAX,40,FLAG,1)
      IF(IMAX.LE.1) GO TO 800
      IF(KMAX.GT.40) KMAX =  40
      CALL XTRCUA(A(WADS),UW,KMAX,40,FLAG,2)
      IF(KMAX.LE.1) GO TO 800
      NMAX = FLD(18,9,A(TADS-1)) / 3
      IF(NMAX.EQ.0) GO TO 80
      IF(NMAX.GT.3) NMAX = 3
      TADS = TADS + FLD(9,9,A(TADS-1)) - 1
      CALL XTRCUA(A(TADS),PT,NMAX,3,FLAG,3)
```

```
   80 CONTINUE
C**   EVALUATE WIND COMPONENTS RELATIVE TO BNL UTM GRID
      OFFSET =  UASTNA(STN)
      DO 100 K = 1,KMAX.
      ANGLE = RADPDG*(D(K) + OFFSET)
      UW(K) = -S(K)*SIN(ANGLE)
      VW(K) =-S(K)*COS(ANGLE)
  100 CONTINUE
C  EVALUATE PRESSURE LEVEL DATA
      IF(FLD(32,1,FLAG).NE.0) GO TO 135
      K = 2
      ZS(1) = UASTNZ(STN)
  105 CONTINUE
      IF(DP(K).LT.0) GO TO 110
      DZ = 62.2*SATVP(TS(K) - DP(K)) / PS(K)
      DZ = DZ / (1.-0.61*DZ)
      TV(K) = TS(K) * (1.0 + 0.61 * DZ)
      GO TO 115
  110 CONTINUE
      TV(K) = TS(K)
  115 CONTINUE
      IF(ZS(K).GT.0.) GO TO 120
      K = K + 1
      GO TO 105
  120 CONTINUE
      IF(K.EQ.2) GO TO 130
      DO 125 I = K,3,-1
      DELTA(I) = ALOG(PS(I) / PS(I-1)) / ALOG(TV(I) / TV(I-1))
      ZS(I-1) = ZS(I) + DELTA(I) * (TV(I) - TV(I-1)) / CONSTA
  125 CONTINUE
  130 CONTINUE
      DZ = ZS(2) - ZS(1)
      TS(1) = TS(2) + GAMA * DZ
      DP(1) = DP(2)
      PS(1) = PS(2) * (TS(1) / TS(2))**(CONSTA / GAMA)
  135 CONTINUE
      K = 2
      DZW = (ZW(2) - ZW(1))
      DO 400 I = 1,IMAX
      IF(DP(I).LT.0.) GO TO 140
      QS(I) = 62.2*SATVP(TS(I) - DP(I))/PS(I)
      QS(I) = QS(I)/(1. - 0.61*QS(I))
      TV(I) = (1. + 0.61*QS(I))*TS(I)
      GO TO 160
  140 CONTINUE
      QS(I) = 0.0
      TV(I) = TS(I)
  160 CONTINUE
      TA(I) = TS(I)*(PO/PS(I))**0.286
```

```
      IF(I.EQ.1) GO TO 300
      IF(TV(I).EQ.TV(I-1)) GO TO 200
      DELTA(I) = ALOG(PS(I)/PS(I-1))/ALOG(TV(I)/TV(I-1))
      IF (ZS(I).LT.0.)
     1ZS(I) = ZS(I-1) - DELTA(I)*(TV(I) - TV(I-1))/CONSTA
      GO TO 300
  200 CONTINUE
      DELTA(I) = 0.0
      IF (ZS(I).LT.0.)
     1ZS(I) = ZS(I-1) - TV(I)*ALOG(PS(I)/PS(I-1))/CONSTA
  300 CONTINUE
      DZ = ZW(K) - ZS(I)
      IF(DZ) 320,340,360
  320 CONTINUE
      IF(K.GE.KMAX) GO TO 500
      K = K+1
      DZW = ZW(K) - ZW(K-1)
      GO TO 300
  340 CONTINUE
      US(I) = UW(K)
      VS(I) = VW(K)
      GO TO 370
  360 CONTINUE
      DZ = DZ/DZW
      US(I) = UW(K) - DZ*(UW(K) - UW(K-1))
      VS(I) = VW(K) - DZ*(VW(K) - VW(K-1))
  370 CONTINUE
      IF(PS(I).LT.PSTD(2)) GO TO 380
      PD(I) = ZS(I) - TSTD(1)*(1.0-(PS(I)/PSTD(1))**(GAMA/CONSTA))/GAMA
      GO TO 400
  380 CONTINUE
      PD(I) = ZS(I) - ZSTD + TSTD(2)*ALOG(PS(I)/PSTD(2))/CONSTA
  400 CONTINUE
      GO TO 600
  500 CONTINUE
      IMAX = I-1
  600 CONTINUE
      IF(NMAX.EQ.0) GO TO 760
      I = 1
      K = 1
  610 CONTINUE
      K = K + 1
      IF(K.GT.IMAX) GO TO 720
      IF(PS(K) - PT(I)) 640,620,610
  620 CONTINUE
      QT(I) = QS(K)
      TB(I) =TV(K)
      TI(I) = TA(K)
      TD(I) = PD(K)
      ZT(I) = ZS(K)
```

```
      IF((UT(I).LT.0.0).OR.(VT(I).LT.0.0)) GO TO 630
      ANGLE = RADPDG*(UT(I) + OFFSET)
      UT(I) = -VT(I) * SIN(ANGLE)
      VT(I) = -VT(I) * COS(ANGLE)
      US(K) = UT(I)
      VS(K) = VT(I)
      GO TO 710
  630 CONTINUE
      UT(I) = US(K)
      VT(I) = VS(K)
      GO TO 710
  640 CONTINUE
      IF(QT(I).LT.0.) GO TO 650
      QT(I) = 62.2 * SATVP(TT(I) - QT(I)) / PT(I)
      QT(I) = QT(I) / (1.0-0.61*QT(I))
      TB(I) = (1.0+0.61*QT(I)) * TT(I)
      GO TO 660
  650 CONTINUE
      QT(I) = 0.0
      TB(I) = TT(I)
  660 CONTINUE
      TI(I) = TT(I) * (PO/PT(I)) ** 0.286
      IF(TB(I).EQ.TV(K-1)) GO TO 670
      DZW = ALOG(PT(I)/PS(K-1)) / ALOG(TB(I) / TV(K-1))
      ZT(I) = ZS(K-1) - DZW * (TB(I) - TV(K-1)) / CONSTA
      GO TO 680
  670 CONTINUE
      ZT(I) = ZS(I-1) - TB(I) * ALOG(PT(I) / PS(K-1)) / CONSTA
  680 CONTINUE
      IF((UT(I).LT.0.0).OR.(VT(I).LT.0.0)) GO TO 690
      ANGLE = RADPDG * (UT(I) + OFFSET)
      UT(I) = -VT(I) * SIN(ANGLE)
      VT(I) = -VT(I) * COS(ANGLE)
      GO TO 700
  690 CONTINUE
      DZ = (ZT(I) - ZS(K-1)) / (ZS(K) - ZS(K-1))
      UT(I) = US(K-1) + (US(K) - US(K-1)) * DZ
      VT(I) = VS(K-1) + (VS(K) - VS(K-1)) * DZ
  700 CONTINUE
      IMAX = IMAX + 1
      IF(IMAX.GT.40) IMAX = 40
      TADS = IMAX - K
      DO 705 J = 1,TADS
      PS(IMAX+1-J) = PS(IMAX-J)
      QS(IMAX+1-J) = QS(IMAX-J)
      TS(IMAX+1-J) = TS(IMAX-J)
      ZS(IMAX+1-J) = ZS(IMAX-J)
      TV(IMAX+1-J) = TV(IMAX-J)
```

```
      TA(IMAX+1-J) = TA(IMAX-J)
      US(IMAX+1-J) = US(IMAX-J)
      VS(IMAX+1-J) = VS(IMAX-J)
      DELTA(IMAX+1-J) = DELTA(IMAX-J)
      PD(IMAX+1-J) = PD(IMAX-J)
  705 CONTINUE
      PS(J) = PT(I)
      QS(J) = QT(I)
      TS(J) = TT(I)
      ZS(J) = ZT(I)
      TV(J) = TB(I)
      TA(J) = TI(I)
      US(J) = UT(I)
      VS(J) = VT(I)
      DELTA(J) = DZW
      PD(J) = TD(I)
  710 CONTINUE
      I = I + 1
      IF(I.LE.NMAX) GO TO 610
      GO TO 760
  720 CONTINUE
      NMAX = I - 1
  760 CONTINUE
      RETURN
  800 CONTINUE
C  INADEQUATE DATA. ELIMINATE OB FROM INVENTORY
      WRITE(6,900) FLAG,IMAX,KMAX,TADS,WADS,
     *             A(TADS-1),A(TADS),A(WADS-1),A(WADS)
  900 FORMAT(' FL,IMX,KMX,TAD,WAD',5I10/
     *       ' A(T-1),A(T),A(W-1),A(W)',4(2X012))
      FLAG = 0
      TLVLMX(STN,RUN) = 0
      WLVLMX(STN,RUN) = 0
      RETURN
      END
```

```
      FUNCTION IDIFTM(I,J)
      M1 = MOD(I,100)
      M2 = MOD(J,100)
      N1 = I/100
      N2 = J/100
      IDIFTM = (N1-N2)*24 + (M1-M2)
      RETURN
        END
      FUNCTION INTRP(X,Y,N,X0)
C
C*****************************************************************************
C*
C*    ROUTINE INTRP (INTERPOLATION) RETURNS A VALUE Y0 INTERPOLATED
C*    FROM A TABLE OF N VALUES X(.),Y(.) USING AN ARGUMENT X0.  THE
C*    TABLE ARGUMENTS X(.) MUST BE MONOTONIC (INCREASING OR DECREASING)
C*
C*****************************************************************************
C
      DIMENSION X(N), Y(N)
C
C**   DETERMINE 2 TABLE ARGUMENTS CLOSEST TO X0
      N1 = N-1
C**   X(.) IS DECREASING
C**   IF X(.) IS DECREASING, BRANCH TO 25
      IF(X(1).GT.X(2)) GO TO 25
C**   X(.) IS INCREASING
      DO 10 I=2,N1
      JJ = I
      IF(X0.LE.X(JJ)) GO TO 50
 10   CONTINUE
      JJ = N
      GO TO 50
C
 25   CONTINUE
      DO 30 I=2,N1
      JJ = I
      IF(X0.GE.X(JJ)) GO TO 50
 30   CONTINUE
      JJ = N
C
 50   J = JJ-1
C**   THE TWO ARGUMENTS CLOSEST TO X0 ARE X(J) AND X(JJ)
      Y0 = ((Y(JJ)-Y(J))/(X(JJ)-X(J)))*(X0-X(J))+Y(J)
      INTRP = Y0
C
      RETURN
      END
```

```
      SUBROUTINE MESMOD
C
C***  THIS ROUTINE OBTAINS WHITE SANDS MESOMODEL DATA STORED
C***  ON FASTRAND FILES
C
      PARAMETER MMBUF=961
      INCLUDE MESAGE
      INCLUDE FILES
      INCLUDE COORD
      INCLUDE DATPRM
      INCLUDE MRCDAT
      INCLUDE MRCIPF
      INCLUDE SCRTCH
C
      DIMENSION DBUF(MMBUF), INDEX(NMMX)
      DIMENSION INDEXT(NMMX), XTMM(NMMX), YTMM(NMMX)
      DIMENSION RMMVAR(LAYERS,NMMX,7)
      EQUIVALENCE (RMMVAR(1,1,1),HMM(1,1))
      DATA XCORN,YCORN /300.0,3670.0/
C
C***  INITIALIZE FOR ACQUISTION OF TRANSPORT LAYER DATA
      IFILE = GRIDF2
      LAYER = 2
C
C***  CALL HEADER FOR MESO DATA TO CORRELATE TIME WITH FASTRAND ADDRESS
C***  DUMMY AT PRESENT
C
      IRUN = 1
C
C***  CALCULATE NMMX INDICES REPRESENTING MESO GRID POINTS CLOSE TO
C***  XPLACE,YPLACE
C***  START BY DETERMINING CLOSEST MESO GRID POINT WHICH IS SW OF
C***  (XPLACE,YPLACE)
      I = (XPLACE-XCORN)/5.0 + 1
      J = (YPLACE-YCORN)/-5.0 + 1
C
C
      IF(I.GT.31.OR.I.LT.0) GO TO 31416
      IF(J.GT.31.OR.J.LT.0) GO TO 31416
      IF(I.GT.29) I=I-1
      IF(I.GT.29) I=I-1
      IF(J.LT.3) J=J+1
      IF(J.LT.3) J=J+1
C
C***  DETERMINE INDICES OF GRID POINTS IN 3-BY-3 SQUARE WITH SW CORNER
C***  (I,J)
      DO 50 L=1,3
      DO 50 K=1,3
      M = K+(L-1)*3
      II = I+(K-1)
      JJ = J-(L-1)
      INDEXT(M) = II+(JJ-1)*31
      XTMM(M) = XCORN+(II-1)*5
      YTMM(M) = YCORN-(JJ)*5
   50 CONTINUE
```

```
C
      CALL CLSSTN(NSTATN,XTMM,YTMM,IDXSTA,STADST,NMMX,100.0)
      DO 55 L=1,NMMX
      K = IDXSTA(L)
      INDEX(L) = INDEXT(K)
      XGRDMM(L) = XTMM(K)
      YGRDMM(L) = YTMM(K)
      RMDIST(L) = STADST(L)
 55   CONTINUE
C
C
C**   CALL IN ELEVATION DATA AND LOAD ZGRDMM
      CALL BLKIN(MMBUF,DBUF(1),1,TRRNFL,ISTS)
C
      DO 60 J=1,NMMX
      K = INDEX(J)
 60   ZGRDMM(J) = DBUF(K)
C
C
C
C**   SET SWITCH FOR PROPER LAYER ACQUISITION
C**   ONLY ONE LAYER PRESENTLY USED
      ISW=1
 100  CONTINUE
C
C**   CALL IN MESO DATA AND LOAD VARIABLES
      NBLK = IMDT(4)-1
      DO 110 IJK=1,7
      NBLK = NBLK+1
      CALL BLKIN(MMBUF,DBUF(1),NBLK,IFILE,ISTS)
      DO 105 J=1,NMMX
      K = INDEX(J)
 105  RMMVAR(LAYER,J,IJK) = DBUF(K)
 110  CONTINUE
      DO 200 J = 1,NMMX
      DUM = .70710678*(AVM(LAYER,J)+AVN(LAYER,J))
      AVN(LAYER,J)=.70710678*(AVN(LAYER,J)-AVM(LAYER,J))
      AVM(LAYER,J) = DUM
      DUM = .70710678*(AVM1(LAYER,J)+AVN1(LAYER,J))
      AVN1(LAYER,J)=.70710678*(AVN1(LAYER,J)-AVM1(LAYER,J))
      AVM1(LAYER,J) = DUM
 200  CONTINUE
```

```
C
C
      L = LAYER
      WRITE(XOTFIL,2000) (I,XGRDMM(I),YGRDMM(I),ZGRDMM(I),HMM(L,I),
     1  AVM(L,I),AVN(L,I),AVM1(L,I),AVN1(L,I),
     2  I=1,NMMX)
C
C**   CHECK WHETHER SURFACE LAYER IS DESIRED
      IF(ISW.EQ.1) RETURN
      ISW = 1
      IFILE = GRIDF1
      LAYER = 1
      GO TO 100
C
31416 NMMNUM = 0
      WRITE(XOTFIL,3000)
3000  FORMAT('   COORDINATES NOT IN MESO GRID REGION ')
 2000 FORMAT('1 MESO DATA'// T2,'GRID POINT',T20,'UTM X',T30,'UTM Y',
     1  T40,'ELEVATION',T53,'HMM',T66,'AVM',T78,'AVN',T89,'AVM1',T100,
     2  'AVN1',T110,'AVUE',T127,'AVVE:'/
     3  (T10,I2,T17,F7.1,T27,F8.1,T41,F8.1,T51,F8.1,T63,F9.1,T76,F8.1,
     4  T87,F8.1,T98,F8.1))
      RETURN
      END
```

```
      SUBROUTINE MESNET
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE MRCGWC
C
C**   THIS SUBROUTINE SEARCHES THE GLOBAL WEATHER CENTRAL DATA
C**   SETS FOR DATA AT FOUR GRID POINTS AT PREDICTION TIMES CLOSE
C**   TO THE UPPER AIR OBSERVATION TIME AND SIMULATION TIME
C
      INCLUDE DATPRM
      INCLUDE MESAGE
      INCLUDE FILES
      INCLUDE GWDATA
      INCLUDE SCRTCH
      INCLUDE GBUFER
      INCLUDE COORD
      INCLUDE MRCDAT
C
      INTEGER GWCBLK
      DATA CRITGW/212.13/
C
C
      GWCBLK = MXBLKG+1
C
C**   ROLL IN GWC HEADER BLOCK
90    CONTINUE
      GWCBLK=GWCBLK-1
      IF(GWCBLK.LE.0) GO TO 7000
      CALL BLKIN(GPTS,GRIDX(1),GWCBLK,GWCOBF,ISTS)
C
C**   FIND CLOSEST FOUR GWC GRID POINTS
      CALL CLSSTN(NSTATN,GRIDX,GRIDY,IDXSTA,STADST,GRDMAX,CRITGW)
      IF(NSTATN.GE.4) NSTATN=4
      IF(NSTATN.LE.0) GO TO 90
C
C**   NEARBY STATIONS EXIST
200   CONTINUE
      K= RUNMAX+1
210   CONTINUE
      K=K-1
      IF(K.LT.1) GO TO 90
      IF(ANLTIM(K).LE.0) GO TO 210
C**   TIME = SIMULATION TIME
C**   ANLTIM(K) = ANALYSIS TIME FOR GWC RUN K
C**   COMPUTE TIME-ANLTIM(K)
      M=IDIFTM(TIME,ANLTIM(K))
C**   IF M.LT.5, RUN K ANALYSIS TIME IS TOO RECENT
C**   SEEK OLDER GWC TIME
      IF(M.LT.5) GO TO 210
C**   IF M.GT.18, GWC DATA IS TOO OLD
      IF(M.GT.18) GO TO 90
C**   IF NO UA DATA EXISTS, GWC UPDATE IS POINTLESS
C**   GET GWC DATA AT SIMULATION ONLY
      IF(NUANUM.EQ.0) GO TO 240
```

```
C**   SET ITR AND GTIME FOR ACQUISITION OF GWC DATA AT UA TIME
      ITR = 1
      GTIME = UPTIME(1)
      WRITE(XQTFIL,1000) GTIME
      GO TO 250
C**   SET ITR AND GTIME FOR ACQUISITION OF GWC DATA AT SIMULATION TIME
  240 ITR = 2
      GTIME = TIME
      WRITE(XQTFIL,1001) GTIME
  250 CONTINUE
C
C**   INITIALIZE COUNTERS
C**   L COUNTS GWC STATIONS TRIED
C**   JN COUNTS GWC STATIONS ACCEPTED
      L=0
      JN=0
  260 CONTINUE
      L=L+1
C**   IF STATIONS ARE EXHAUSTED, SETTLE FOR WHAT YOU GOT
      IF(L.GT.NSTATN) GO TO 310
      I=IDXSTA(L)
C**   OBTAIN RELEVANT GWC DATA
      CALL GWCDTA(I,GRDBLK(K),IFLAG)
C**   IF DATA IS INCOMPLETE, SKIP IT
      IF(IFLAG.EQ.0) GO TO 260
C**   BUMP COUNTER
      JN=JN+1
C**   LOAD DATA
      IGGNUM(ITR,JN)=I
      GDIST(ITR,JN) = STADST(L)
      XGRD(ITR,JN)=GRIDX(I)
      YGRD(ITR,JN)=GRIDY(I)
      ZGRD(ITR,JN)=0.0
      IFLAGG(ITR,JN)=IFLAG
      GWTIME(ITR,JN)=GTIME
      DO 300 IS=1,4
      PGW(IS,ITR,JN)=P(IS)
      QGW(IS,ITR,JN)=Q(IS)
      TGW(IS,ITR,JN)=T(IS)
      UGW(IS,ITR,JN)=U(IS)
      VGW(IS,ITR,JN)=V(IS)
      ZZGW(IS,ITR,JN) = Z(IS)
  300 CONTINUE
C
C**   PRINT DATA
```

```
      WRITE(XOTFIL,9002)GWCBLK,K,I,GRIDX(I),GRIDY(I),STADST(L),TIME,
     1 ANLTIM(K),GTIME
      WRITE(XOTFIL,9004)
      DO 305 I=1,4
      WRITE(XOTFIL,9005)I,P(I),Q(I),T(I),U(I),V(I),Z(I)
  305 CONTINUE
C
C
C**   IF LESS THAN NGWX STATIONS FOUND, KEEP LOOKING
      IF(JN.LT.NGWX) GO TO 260
  310 CONTINUE
      NGWNUM(ITR)=JN
      IF(ITR.EQ.1) GO TO 240
C
C
      WRITE(XOTFIL,9000)
      RETURN
C
 7000 CONTINUE
      WRITE(XOTFIL,9000)
      WRITE(XOTFIL,9010)
      RETURN
C
 1000 FORMAT('1GWC DATA FOR GWC ANALYSIS TIME=', I8)
 1001 FORMAT('1GWC DATA FOR GWC PREDICTION TIME=', I8)
 9000 FORMAT(///1X,'      *****  GWC DATA PROCESSED ***')
 9002 FORMAT(/////' HEADER DATA'//T2,'BLOCK',T16,'RUN INDEX',
     M     T31,'STA INDEX',T49,
     1    'UTM X',T64,'UTM Y',T76,'DISTANCE',T91,'SIM TIME',
     2    T108,'UATIME', T121,'GWC TIME'//
     3    T2,I5,T20,I5,T35,I5,T46,3(F8.2,7X),2(I8,7X),I8)
 9004 FORMAT(//' PREDICTION DATA'//T9,'PRESSURE',T24,'SPECIFIC',
     M   T38,'TEMPERATURE',T59,
     1  'UWIND',T75,'VWIND',T87,'ELEVATION'/
     2  T9,'(PASCALS)',T24,'HUMIDITY',T38,'(DEGREES K)',
     3  T60,'(M/S)',T76,'(M/S)',T93,'(M)'/)
 9005 FORMAT(T3,I2,T8,F10.1,T23,E11.4,T41,F8.2,T54,F11.2,T70,F11.2,
     1  T85,F11.2)
 9010 FORMAT(1X,' ALL DATA BLOCKS SEARCHED   *** ERROR  ***')
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE MRCSFC
C
C**   THIS ROUTINE ACCESSES THE SURFACE DATA AND LOADS DATA FOR A
C**   MAXIMUM OF NSFX STATIONS INTO LABELED COMMON /MRCDAT/
C
      INCLUDE DATPRM
      INCLUDE MESAGE
      INCLUDE SCRTCH
      INCLUDE MRCIPF
      INCLUDE FILES
      INCLUDE SFDATA
      INCLUDE SBUFER
      INCLUDE MRCDAT
      INCLUDE COORD
C
      INTEGER SRTIME
      DIMENSION NH(1)
      DATA CRIDEC/35.0/,CRITSF/150.0/
C
C
      WRITE(XOTFIL,1)
C
C**   INITIALIZE COUNTERS
C**   IX = SIMULATION TIME - SURFACE STATION TIME
C**   JN = COUNTER FOR STATIONS FOUND WITH ACCEPTABLE DATA
C**   SRTIME = SURFACE STATION TIME USED AS CALLING ARGUMENT
      IX = 0
      JN = 0
      SRTIME = TIME
C**   LOAD SURFACE DATA HEADER
      CALL BLKIN(SOBS,SFCSTN,1,SFCFIL,ISTS)
C
  150 CONTINUE
C**   FIND CLOSEST STATIONS WITHIN CRITSF KM.
      CALL CLSSTN(NSTATN,SFSTNX,SFSTNY,IDXSTA,STADST,STNMAX,CRITSF)
C**   BEGIN LOOP OVER ALL STATIONS WITHIN CRITSF KM OF WIND MODEL
C**   SITE
C**   L = COUNTER FOR LIST OF NEARBY STATIONS
      L = 0
260   CONTINUE
      L = L+1
C**   IF LIST OF POSSIBLE STATIONS IS EXHAUSTED. TRY LATER TIME
      IF(L.GT.NSTATN) GO TO 5000
C
      I = IDXSTA(L)
C**   OBTAIN DATA FOR STATION I AT TIME SRTIME
  265 CALL SFCDTA(SRTIME,I,IFLAG)
      IF(IFLAG.EQ.0) GO TO 260
      LOGIC = 0
C**   DISQUALIFY READING IF VITAL DATA IS MISSING
      IF(TS.LE.0.0) LOGIC=1
      IF(WS.LE.0.0) LOGIC=1
      IF(WD.LT.0.0.OR.WD.GT.360.0) LOGIC=1
C**   IF VITAL DATA IS PRESENT. PROCEED TO NEXT TEST
      IF(LOGIC.NE.1) GO TO 266
```

```
C**   IFLAG.GE.16 MEANS MORE DATA AVAILABLE.  REPEAT CALL TO SFCDTA
      IF(IFLAG.GE.16) GO TO 265
      GO TO 260
  266 CONTINUE
C**   DETERMINE WHETHER THIS STATION HAS BEEN USED ALREADY
C**   IF SO, DON'T USE TWICE.  TR Y OTHER STATIONS
      IF(JN.EQ.0) GO TO 280
      DO 270 IJK=1,JN
      IF(SFCSTN(I).EQ.STSNUM(IJK)) GO TO 260
  270 CONTINUE
  280 CONTINUE
C
C**   SURFACE DATA WAS FOUND.  BUMP COUNTER, PRINT DATA, LOAD MRCDAT
      JN = JN+1
      WRITE(XOTFIL,9004)I,SFCSTN(I),SFSTNX(I),SFSTNY(I),
     1  SFSTNZ(I),SRTIME
      NSFNUM = JN
      WRITE(XOTFIL,9005) WD,PA,NH,WS,TSC,CL,WG,PW,H,VV,CM,SLP,CH,
      WRITE(XOTFIL,9006) BT,WW(1),NS,TS,WW(2),TD,WW(3),K,AS,WW(4),
     1       ,CT,HSHS,STADST(L)
      SDIST(JN) = STADST(L)
      STSNUM(JN) = SFCSTN(I)
      XSTS(JN) = SFSTNX(I)
      YSTS(JN) = SFSTNY(I)
      ZSTS(JN) = SFSTNZ(I)
      IFLAGS(JN) = IFLAG
      SFTIME(JN) = SRTIME
      WDSF(JN) =  WD
      WSSF(JN) = WS
      WGSF(JN) = WG
      SLPSF(JN) = SLP
      BTSF(JN) = BT
      TSSF(JN) = TS
      TDSF(JN) = TD
      ASSF(JN) = AS
      PASF(JN) = PA
      TSCSF(JN) = TSC
      PWSF(JN) = PW
      VVSF(JN) = VV
      DO 300 K = 1,4
      WWSF(K,JN) = WW(K)
300   CONTINUE
      DO 310 K = 1,9
      ICLOUD(K,JN) = NH(K)
310   CONTINUE
C
```

```
C**   IF NSFX STATIONS HAVE BEEN FOUND. QUIT AND BE HAPPY
      IF(NSFNUM.GE.NSFX) GO TO 5100
C**   IF NOT. TRY MORE STATIONS
      GO TO 260
5000  CONTINUE
C
C
C**   TRY DATA ONE HOUR OLDER
      IX = IX+1
      SRTIME = TIME-IX
C**   MAKE DISTANCE REQUIREMENT MORE STRINGENT
      CRITSF = CRITSF-CRIDEC
C**   IF DATA IS TOO OLD. QUIT
      IF(IX.LE.IMDT(2)) GO TO 150
C
C
5100  CONTINUE
      WRITE(XOTFIL,9020) TIME
      WRITE(XOTFIL,9000) NSFNUM
 1    FORMAT('1SURFACE STATION DATA'//' VARIABLE NAMES='/
     1  ' WD.PA.NH.WS.TSC.CL.WG.PW.H.VV.CM.SLP.CH.BT.WW(1),',
     2  'NS.TS.WW(2).TD.WW(3).K.AS.WW(4).CT.HSHS'//
     3  ' FOR DESCRIPTION OF VARIABLES SEE COMMON BLOCK /SFDATA/'/)
9020  FORMAT(1X,'SURFACE DATA EXHAUSTED FOR TIME =' I8)
9004  FORMAT(////' HEADER DATA'/
     1  T42,'STATION INDEX',T61,'STATION ID',T78,'UTM X',T90,'UTM Y',
     2  T102,'UTM Z',T113,'SURFACE OBS TIME'//
     3  T52,I3,T61,I10,T74,F9.3,T86,F9.3,T98,F9.3,T120,I9///)
9005  FORMAT(' OBSERVATION DATA'//
     M  T7,'WIND   DIRECTION',T24,'=',T26,F10.2,T37,'(DEG)',
     M  T51,'6 HR PRECIP',
     M  T68,'=',T70,F10.2,T96,'LOW CLOUD COVER <',T119,'=',T121,I6/
     M  T14,'SPEED',T24,'=',T26,F10.2,T37,'(M/S)',T51,'TOTAL SKY COVER',
     M  T68,'=',T70,I10,T96,'TYPE LOW CLOUDS',T119,'=',T121,I6/
     M  T14,'GUSTS',T24,'=',T26,F10.2,T51,'PAST WEATHER',
     M  T68,'=',T70,I10,T96,'HEIGHT LOW CLOUDS',T119,'=',T121,I6/
     M  T51,'VISIBILITY',
     M  T68,'=',T70,I10,T81,'(M)',T96,'TYPE MIDDLE CLOUDS',
     M  T119,'=',T121,I6/
     M  T7,'PRESSURE',T24,'=',T26,F10.1,T37,'(PASCALS)',
     M  T51,'PRESENT WEATHER',
     M  T96,'TYPE HIGH CLOUD',T119,'=',T121,I6)
9006   FORMAT(1X,T7,'BARO TEND',T24,'=',T26,I10,  T57,'WW(1)',
     M  T68,'=',T70,I10,T96,'AMOUNT CLOUDS',T119,'=',T121,I6/
     M  T7,'TEMPERATURE',T24,'=',T26,F10.2,T37,'(DEG K)',T57,'WW(2)',
     M  T68,'=',T70,I10/
     M  T7,'DEW PT DEPRESS',T24,'=',T26,F10.2,T37,'(DEG K)',
     M  T57,'WW(3)',
     M  T68,'=',T70,I10,T96,'HEIGHT CLASSIFICATION',T119,'=',T121,I6/
     M  T7,'ALTIM SETTING',T24,'=',T26,F10.2,T37,'(INCHES HG)',
     M  T57,'WW(4)',
     M  T68,'=',T70,I10,T96,'TYPE CLOUD',T119,'=',T121,I6/
     M  T96,'HEIGHT TO BASE OF CLD',T119,'=',T121,I6//
     M    T2,'DISTANCE TO SIMULATION POINT = ',F10.2,' KM'/////)
9000  FORMAT(1X,'SURFACE DATA PROCESSED ',I5,' STATIONS FOUND WITH'
     1 ,' DATA'//)
C
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE MRCUA
C
C**   THIS SUBROUTINE SEARCHES THE UPPER AIR DATA SETS FOR DATA AT FOUR
C**   STATIONS SUFFICIENTLY CLOSE TO THE TIME AND PLACE OF THE SURFACE
C**   WIND ESTIMATION.
C
      INCLUDE DATPRM
      INCLUDE MRCIPF
      INCLUDE MESAGE
      INCLUDE COORD
      INCLUDE SCRTCH
      INCLUDE FILES
      INCLUDE MRCDAT
      INCLUDE OBUFER
      INCLUDE TDATA
      INCLUDE WDATA
C
      INTEGER UABLK
C
      DATA CRITUA,OBUFRN/250.0,431/
C
C**   INITIALIZE COUNTERS
C**   ICOUNT COUNTS THE NUMBER OF STATIONS FOUND WITH ACCEPTABLE DATA
C**   AND LOADED INTO COMMON /MRCDAT/ FOR USE BY DATANL.
C**   UABLK INDEXES THE UPPER AIR HEADER BLOCK BEING SEARCHED.
      ICOUNT = 0
      UABLK  = MXBLKU+1
C
C**   LOOP OVER HEADER BLOCKS
   50 UABLK = UABLK-1
      IF(UABLK.LE.0) GO TO 5000
      CALL BLKIN(UOBS,UASTNS(1),UABLK,OBFILE,ISTS)
      IF(UATIME(1).GT.TIME) GO TO 50
      IDIFF = IDIFTM(TIME,UATIME(1))
      IF(IDIFF.GE.24) GO TO 5000
C
C**   THE BLOCK OF UA DATA FOR THE 24 HOUR PERIOD CONTAINING
C**   SIMULATION TIME HAS BEEN FOUND
C**   FIND THE MOST RECENT RUN IN THIS BLOCK
      IRUNMX = 1
      DO 60 I=2,RU
      IF(UATIME(I).GT.TIME) GO TO 70
      IRUNMX = I
   60 CONTINUE
   70 CONTINUE
C
C**   OBTAIN LIST OF STATIONS SUFFICIENTLY CLOSE TO SITE
      CALL CLSSTN(NSTATN,UASTNX,UASTNY,IDXSTA,STADST,STNMAX,CRITUA)
C**   IF NO STATIONS EXIST, QUIT
      IF(NSTATN.EQ.0) GO TO 5000
C**   RECENT DATA FROM NEARBY STATIONS HAS BEEN LOCATED
C**   ACQUIRE DATA
```

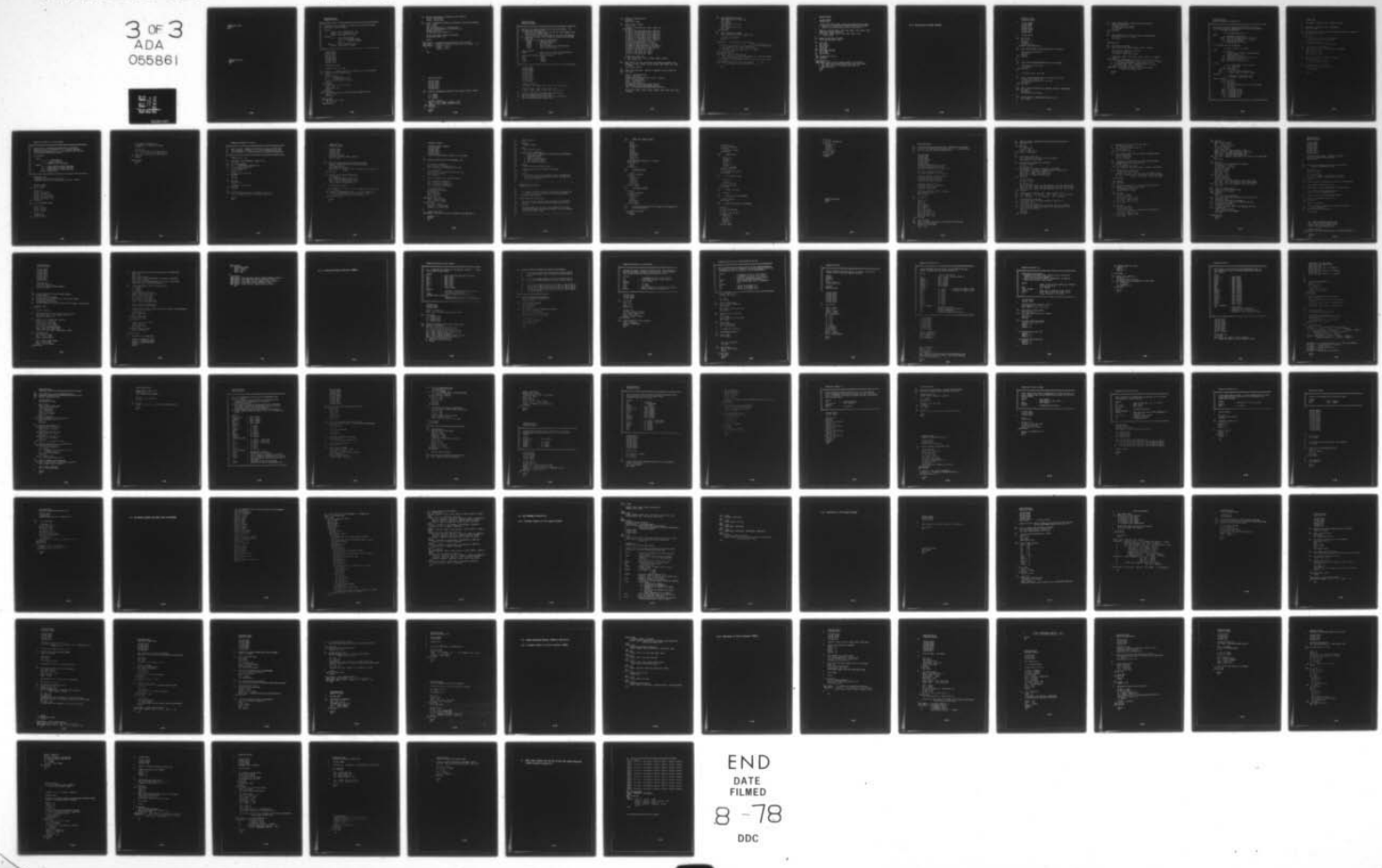
```
C**   LOOP OVER ACCEPTABLE TIMES
      DO 200 I=1,IRUNMX
      IRUN = IRUNMX+1-I
      IDIFF = IDIFTM(TIME,UATIME(IRUN))
      IF(IDIFF.GE.12) GO TO 5000
C**   AS AN OPTION, DATA FROM TIMES JUST PRECEDING AND JUST
C**   FOLLOWING SIMULATION TIME MAY BE SOUGHT AND AVERAGED.
C**   THE DEFAULT PROCEDURE IS TO USE THE REAL-TIME VALUES ONLY
C**   SET THE TIME AVERAGING FACTORS
      TMP = IDIFF/12.0
      F2 = TMP
      F1 = 1 - F2
C**   LOOP OVER NEARBY STATIONS
      DO 190 J=1,NSTATN
C**   INITIALIZE FOR ACQUISITION OF REAL-TIME DATA
      JRUN = IRUN
      IPT = 100
      IW = 100
C**   SET STATION INDEX
      INDEX = IDXSTA(J)
C
C**   THIS IS THE ENTRY POINT FOR THE TIME 'SANDWICH'
  100 CONTINUE
C**   SET NUMBER OF TEMPERATURE AND WIND READINGS FOR THIS STATION AND TIME
      IMAX = TLVLMX(INDEX,JRUN)
      KMAX = WLVLMX(INDEX,JRUN)
C**   GET MINIMUM NUMBER OF READINGS FOR THE TWO DATA SETS
      IPT = MIN0(IPT,IMAX)
      IW = MIN0(IW,KMAX)
C**   IF INSUFFICIENT READINGS ARE AVAILABLE, PREPARE TO SKIP
      IF(IPT.EQ.0.OR.IW.EQ.0) GO TO 103
      IF(IPT.GT.NLVL) IPT=NLVL
      IF(IW.GT.NLVL) IW=NLVL
C**   OBTAIN UA DATA FOR INDICATED STATION AND RUN
      CALL UADATA(INDEX,JRUN,IFLAG)
      IF(IFLAG.NE.0) GO TO 104
C**   DATA IS INCOMPLETE.  SKIP THIS STATION; BUT RESET ICOUNT IF NECESSARY
  103 IF(JRUN.EQ.IRUN) GO TO 190
      ICOUNT = ICOUNT-1
      GO TO 190
C
  104 CONTINUE
C**   BRANCH IF DATA IS TO BE TIME AVERAGED
      IF(JRUN.NE.IRUN) GO TO 120
C
      ICOUNT = ICOUNT + 1
      IC = ICOUNT
```

```
C**   STORE DATA IN COMMON /MRCDAT/
C**   PRESSURE AND TEMPERATURE READINGS
      NTR(ICOUNT)= IPT
      DO 110 K=1,IPT
      PRES(K,ICOUNT) = PS(K)
      TEMP(K,ICOUNT) = TS(K)
      QHUM(K,ICOUNT) = QS(K)
      ZMSL(K,ICOUNT) = ZS(K)
      VRTEMP(K,ICOUNT) = TV(K)
      POTEMP(K,ICOUNT) = TA(K)
      UWIND(K,ICOUNT) = US(K)
      VWIND(K,ICOUNT) = VS(K)
  110 CONTINUE
C
C**   WIND READINGS
      NWR(ICOUNT) = IW
      DO 115 K=1,IW
      UVEL(K,ICOUNT) = UW(K)
      VVEL(K,ICOUNT) = VW(K)
      ZWDMSL(K,ICOUNT) = ZW(K)
  115 CONTINUE
C
C**   IF TIME AVERAGED RESULTS ARE NOT DESIRED, SKIP AHEAD
      IF(IMDT(1).NE.1) GO TO 155
C**   TIME AVERAGING REQUESTED.  SET JRUN TO NEXT RUN AND REENTER DATA
C**   ACQUISITION LOOP
      JRUN = IRUN+1
      GO TO 10
C
  120 CONTINUE
C**   THIS CODE TIME AVERAGES TWO SETS OF UA DATA
      DO 130 K=1,IPT
      PR = PRES(K,IC)
      TEMP(K,IC) = F1*TEMP(K,IC)+F2*INTRP(PS,TS,IPT,PR)
      QHUM(K,IC) = F1*QHUM(K,IC)+F2*INTRP(PS,QS,IPT,PR)
      ZMSL(K,IC) = F1*ZMSL(K,IC)+F2*INTRP(PS,ZS,IPT,PR)
      VRTEMP(K,IC) = F1*VRTEMP(K,IC)+F2*INTRP(PS,TV,IPT,PR)
      POTEMP(K,IC) = F1*POTEMP(K,IC)+F2*INTRP(PS,TA,IPT,PR)
      UWIND(K,IC) = F1*UWIND(K,IC)+F2*INTRP(PS,US,IPT,PR)
      VWIND(K,IC) = F1*VWIND(K,IC)+F2*INTRP(PS,VS,IPT,PR)
  130 CONTINUE
C
C**   WIND READINGS
C
      NWR(IC) = IW
      DO 135 K=1,IW
      ZP = ZWDMSL(K,IC)
      UVEL(K,IC) = F1*UVEL(K,IC)+F2*INTRP(ZW,UW,IW,ZP)
      VVEL(K,IC) = F1*VVEL(K,IC)+F2*INTRP(ZW,VW,IW,ZP)
  135 CONTINUE
C
  155 CONTINUE
C
```

```
      UPTIME(ICOUNT) = UATIME(IRUN)
      IF(IMDT(1).EQ.1) UPTIME(ICOUNT)=TIME
      STUNUM(ICOUNT) = UASTNS(INDEX)
      XSTA(ICOUNT) = UASTNX(INDEX)
      YSTA(ICOUNT) = UASTNY(INDEX)
      ZSTA(ICOUNT) = UASTNZ(INDEX)
      IFLAGU(ICOUNT) = IFLAG
      UDIST(ICOUNT) = STADST(J)
C
C**   PRINT DATA
      WRITE(XOTFIL,1000) UABLK,IRUN,UASTNS(INDEX),
     1  UASTNX(INDEX),UASTNY(INDEX),UASTNZ(INDEX),UPTIME(ICOUNT),
     2     IFLAG,INDEX
      N = MIN0(NLVL,IPT,IW)
      DO 160 K=1,N
  160 WRITE(XOTFIL,2000) PRES(K,IC),TEMP(K,IC),QHUM(K,IC),
     1      ZMSL(K,IC),UWIND(K,IC),VWIND(K,IC),VRTEMP(K,IC),
     2      POTEMP(K,IC)
      WRITE(XOTFIL,3000)
      DO 170 K=1,N
  170 WRITE(XOTFIL,4000) UVEL(K,IC),VVEL(K,IC),ZWDMSL(K,IC)
      WRITE(XOTFIL,4005) STADST(J)
C**   IF NUAX STATIONS HAVE BEEN FOUND, QUIT AND BE HAPPY
      IF(ICOUNT.GE.NUAX) GO TO 5000
  190 CONTINUE
  200 CONTINUE
C
      IF(ICOUNT.LT.NUAX) GO TO 50
 5000 NUANUM = ICOUNT
      NUAXX=NUAX
      WRITE(XOTFIL,6000) NUANUM,NUAXX
      RETURN
C
 1000 FORMAT(1H1, 'UPPER AIR DATA FOUND USING HEADER BLOCK NUMBER',I3//
     1  T7,'RUN INDEX     STATION NUMBER     UTM X     UTM Y',
     2  '     ELEVATION (METERS)     OBSERVATION TIME     DATA FLAG',
     3  '  STN INDEX'/T13,I3,T29,I6,T37,F8.1,T47,F8.1,T59,F10.2,T93,I6,
     4  T104,I9,I11///' PRESSURE, TEMPERATURE, AND WIND READINGS'//
     5  T7,' PRESSURE     TEMPERATURE',10X,'SPECIFIC      OBSERVATION',
     6   '     WIND VELOCITY     WIND VELOCITY',T107,'VIRTUAL',
     7  T120,'POTENTIAL'/
     8  T7,'(PASCALS)     (DEGREES K)',10X,'HUMIDITY     HEIGHT (P,T'
     9  ,')     U COMPONENT       V COMPONENT',
     T  T107,'TEMP', T120,'TEMP'//)
 2000 FORMAT(T6,F10.1,T26,F6.2,T40,1PE10.4,T60,0PF7.1,T76,F7.2,T94,
     1  F7.2,T107,F7.2,T120,F7.2)
 3000 FORMAT(//' SIGNIFICANT WIND LEVELS'//
     1  T7, 'WIND VELOCITY     WIND VELOCITY', T45,'OBSERVATION'/
     2  T7, 'U COMPONENT       V COMPONENT',T45,'HEIGHT')
 4000 FORMAT(T7,F7.2,T26,F7.2,T45,F7.1)
 4005 FORMAT(///' DISTANCE TO SITE = ',F8.2,' KM')
 6000 FORMAT(///' DATA SEARCH COMPLETE'/,1X,I3,' RUN(S) WITH '
     1  ,'ACCEPTABLE DATA FOUND.  ',I3,' RUNS WERE SOUGHT')
C
      END
```

AD-A055 861 MISSION RESEARCH CORP SANTA BARBARA CALIF F/G 4/2
PHYSICALLY-BASED HIGH RESOLUTION SURFACE WIND AND TEMPERATURE A--ETC(U).
MAR 78 J A BALL, S A JOHNSON DAEA18-77-C-0043
UNCLASSIFIED MRC-R-7731-1-278 ERADCOM/ASL-CR-78-0043-1 NL

3 OF 3
ADA
055861



END
DATE
FILMED
8-78
DDC

```
      SUBROUTINE PLTPRG
      RETURN
      END
```

```
      SUBROUTINE MIXLYR
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE TERACQ
C
C*****************************************************************************
C*
C*    SUBROUTINE TERACQ ACQUIRES MICRO TERRAIN AND SURFACE ROUGHNESS
C*    DATA USED IN ROUTINE WINDEX.
C*
C*    INPUTS:
C*           XPLACE = UTM X-COORDINATE OF SITE
C*           YPLACE = UTM Y-COORDINATE OF SITE
C*           MTRBLK = DATA ACQUISITION FLAG
C*    OUTPUTS:
C*           H(...) = MICRO TERRAIN ARRAY
C*                    H(1,1) IS SOUTHWEST CORNER
C*                    H(I,ILL) IS NORTHWEST CORNER
C*                    ETC
C*        ROUGH(...) = SURFACE ROUGHNESS ARRAY
C*            DELTER = GRID SPACING (METERS)
C*
C*****************************************************************************
C
      INCLUDE GRDPRM
      INCLUDE MESAGE
      INCLUDE FILES
      INCLUDE COORD
      INCLUDE MRCIPF
      INCLUDE TERDAT
      INCLUDE TBLHDR
      INCLUDE INTLYZ
C
      DIMENSION DST(IDIM)
C
      IF(MTRBLK.GT.0) GO TO 200
C
C**   MTRBLK = 0:  SEARCH DIRECTORY FOR TERRAIN BLOCK WITH SOUTHWEST
C**   CORNER CLOSEST TO XPLACE,YPLACE
      IMIN = 1
      DO 100 I=1,NNBLK
      DST(I) = (XPLACE-SWUTMX(I))**2
     1         +(YPLACE-SWUTMY(I))**2
      IF(DST(I).GT.DST(IMIN)) GO TO 100
      IMIN = I
  100 CONTINUE
      IF(DST(IMIN).LT.10.) GO TO 150
      IHIERC = 0
      WRITE(XOTFIL,1000)
      RETURN
  150 CONTINUE
C**   IMIN IDENTIFIES BLOCKS CONTAINING DATA CLOSEST TO SITE
      MTRBLK = IMIN
      GO TO 300
C
  200 CONTINUE
C**   USE MTRBLK TO SET IMIN
      IMIN = MTRBLK
  300 CONTINUE
```

```
C**   DEFINE XPLACE,YPLACE TO AGREE WITH DATA LOCATION
      XPLACE = SWUTMX(IMIN)
      YPLACE = SWUTMY(IMIN)
C
C**   BLOCK IN APPROPRIATE TERRAIN AND ROUGHNESS DATA AND GRID SPACING
      ITER = 2*IMIN
      IRUF = ITER+1
      CALL BLKIN(NFOUT,H(1,1),ITER,MICTRF,IST)
      CALL BLKIN(NFOUT,ROUGH(1,1),IRUF,MICTRF,IST)
      DELTER = DELH(IMIN)
      IF(ZPLACE.EQ.0.0) ZPLACE=H(1,1)
C
      WRITE(XOTFIL,3000) XPLACE,YPLACE,ZPLACE
      WRITE(XOTFIL,2000) IMIN
C
      RETURN
C
 1000 FORMAT(//' NO TERRAIN DATA SUFFICIENTLY CLOSE TO SITE')
 2000 FORMAT(//' TERRAIN DATA TAKEN FROM DIRECTORY SITE NUMBER ',I3//)
 3000 FORMAT(// ' XPLACE = ',F8.2/
     1          ' YPLACE = ',F8.2/
     2          ' ZPLACE = ',F8.2)
C
      END
```

```
C
      SUBROUTINE TMPPRF
C
      INCLUDE MESAGE
      INCLUDE FILES
      INCLUDE INTLYZ
      INCLUDE COORD
      INCLUDE COEF
      INCLUDE DATFLG
C
C**   POTENTIAL TEMPERATURE PROFILE AT SITE GIVEN BY TPRO(L),ZPRO(L)
C
      XX = XPLACE
      YY = YPLACE
      ZZ = ZPLACE
C
      DO 250 L=1,LEVELS
      TPRO(L) = AA(L,1)+BB(L,1)*XX+CC(L,1)*YY
      ZPRO(L) = AA(L,4)+BB(L,4)*XX+CC(L,4)*YY
  250 CONTINUE
      NVFLAG(1) = 1
C
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE WNDMGR
C
C*****************************************************************************
C*
C*    WNDMGR IS THE OVERALL MANAGER FOR THE MICRO WIND MODEL.  ITS
C*    FUNCTIONS ARE THE FOLLOWING:
C*    1.  ACQUISITION OF HEADER INFORMATION FOR THE MICRO TERRAIN INPUT
C*        AND MICRO WIND OUTPUT FILES
C*    2.  INPUT OF FLAG DATA GIVING OPTIONS FOR THE RUN TO BE EXECUTED
C*    3.  CONVERSION OF LOCAL TIME TO GREENWICH PACKED-JULIAN-DAY-AND-
C*        HOUR TIME
C*    4.  MANAGEMENT OF ITS PRINCIPAL SUBORDINATES:
C*          TERACQ      -   TERRAIN ACQUISITION
C*          DTBMGR      -   DATA BASE SEARCH
C*          DATANL      -   DATA ANALYSIS
C*          DRIVRS      -   CHECKS DRIVERS AGAINST PREVIOUS RUNS
C*          WINDEX      -   MICRO WIND SIMULATION
C*
C*    LOCAL VARIABLES:
C*    MONTHS            -   ARRAY OF DAYS-IN-MONTH FOR CONVERSION OF
C*                          LOCAL TIMES TO JULIAN TIMES
C*    LOCATION OF IMPORTANT COMMON VARIABLES:
C*    LTIME             -   /COORD/
C*    TIME              -   /COORD/
C*    IHIERC            -   /MRCIPF/
C*    ILASTR            -   /MRCIPF/
C*
C*****************************************************************************
C
      INCLUDE DATPRM
      INCLUDE GRDPRM
      INCLUDE MESAGE
      INCLUDE FILES
      INCLUDE COORD
      INCLUDE MRCIPF
      INCLUDE TERDAT
      INCLUDE INTLYZ
      INCLUDE MRCHDR
      INCLUDE TBLHDR
C
      DIMENSION MONTHS(12)
      DATA MONTHS /0,31,59,90,120,151,181,212,243,273,304,334/
C
      NAMELIST /FLAGS/  IDAT, IMDT, ILASTR, NTST,
     1  IFLUX, IHIERC, MTRBLK, IRELAX, ILE, ILW, JLN, JLS
C
C**   BLOCK IN DIRECTORY OF ARCHIVED MICRO TERRAIN DATA
      CALL BLKIN(THDRSZ,NNBLK,1,MICTRF,IST)
C**   BLOCK IN DIRECTORY OF ARCHIVED MICRO WIND SIMULATIONS
      CALL BLKIN(HDRSIZ,RCOUNT,1,MOTFIL,IST)
C
C
```

```
C**   ENTRY FOR SUCCESIVE RUNS
    1 CONTINUE
      READ(XINFIL,FLAGS)
C
C**   SUPPLY DEFAULT VALUES
      DO 20 I=1,3
      IF(IDAT(1).NE.0.AND.IDAT(I).NE.1) IDAT(I)=0
   20 CONTINUE
      IF(IDAT(4).NE.0.AND.IDAT(4).NE.1) IDAT(4)=1
      IF(IDAT(6).NE.0.AND.IDAT(6).NE.1) IDAT(6)=1
      IF(IMDT(1).LT.0.OR.IMDT(1).GT.1) IMDT(1)=0
      IF(IMDT(2).LT.0.OR.IMDT(2).GT.3) IMDT(2)=3
      IF(IMDT(3).LT.0.OR.IMDT(3).GT.6) IMDT(3)=6
      IF(IMDT(4).LT.1.OR.IMDT(4).GT.8) IMDT(4)=8
      IF(IMDT(6).LT.-1.OR.IMDT(6).GT.1) IMDT(6)=1
      IF(NTST.NE.0.AND.NTST.NE.1) NTST=0
      IF(IFLUX.GT.4.AND.IFLUX.LT.1) IFLUX=2
      IF(IHIERC.GT.3.AND.IHIERC.LT.0) IHIERC=0
      IF(MTRBLK.GT.NMBLK.AND.MTRBLK.LT.0) MTRBLK=0
      IF(IRELAX.LT.0.OR.IRELAX.GT.100) IRELAX=5
      IF(ILE.LT.1.OR.ILE.GT.ILL) ILE=ILL
      IF(ILW.LT.1.OR.ILW.GT.ILL) ILW=1
      IF(JLN.LT.1.OR.JLN.GT.JLL) JLN=JLL
      IF(JLS.LT.1.OR.JLS.GT.JLL) JLS=1
C
      WRITE(XOTFIL,1000) IDAT,
     1  IMDT, ILASTR, NTST, IFLUX, IHIERC, MTRBLK, IRELAX,
     2   ILE,ILW,JLN,JLS
C
C**   READ SPATIAL AND TIME COORDINATES AND DETERMINE GREENWICH TIME
      NAMELIST /TIMPLA/ XPLACE, YPLACE, ZPLACE, YEAR, MONTH, DAY, HOUR,
     1   MINUTE
      READ (XINFIL,TIMPLA)
C
C**   INPUT TIMES ARE LOCAL.  CONVERT TO GREENWICH JULIAN PACKED DAY
C**   AND HOUR
C
      JULIAN = MONTHS(MONTH)+DAY
      LEAPYR = MOD(YEAR,4)
      IF(LEAPYR.EQ.0.AND.MONTH.GT.2) JULIAN = JULIAN+1
      LTIME = JULIAN*100+HOUR
      IDIFF = (HOUR+TIMDIF)/24
      JULIAN = JULIAN+IDIFF
      IF(LPYEAR.EQ.0.AND.JULIAN.EQ.367) JULIAN=1
      IF(LPYEAR.EQ.1.AND.JULIAN.EQ.366) JULIAN=1
      TIME = (JULIAN+IDIFF)*100+(HOUR+TIMDIF-IDIFF*24)
C
      WRITE(XOTFIL,3000) XPLACE, YPLACE, ZPLACE, YEAR, MONTH, DAY, HOUR,
     1  MINUTE
C
```

```
C**   CALL SUBORDINATE ROUTINES
      WRITE(XOTFIL,4000) LTIME, TIME
      CALL TERACQ
      CALL DTBMGR
      IF(IHIERC.EQ.0) GO TO 40
      CALL DRIVRS
      IF(IHIERC.EQ.0) GO TO 40
      CALL WINDEX
   40 CONTINUE
C
C**   OUTPUT UPDATED PLOT HEADER
      CALL BLKOUT(HDRSIZ,RCOUNT,1,MOTFIL,IST)
C
      IF(ILASTR.EQ.1) RETURN
      GO TO 1
C
 1000 FORMAT(//' INPUT FLAGS TO MRCIPF COMMON'/
     1                                        T35,'-------IDAT-------',
     2  T59,'-------IMDT-------',T82,'ILASTR',T94,'NTST',T104,'IFLUX',
     3  T115,'IHIERC',T126,'MTPBLK'/T35,6I3,T59,
     4  6I3,T83,I5,T94,I4,T104,I5,T115,I5,T127,I5//
     5  T35,'IRELAX ILE  ILW  JLN  JLS        '/T35,
     6     I6,I4,3I5,I8//)
 2000 FORMAT(3I10, 5I5)
 3000 FORMAT(' INPUT TO COORD COMMON'/
     1  T35,'XPLACE',T47,'YPLACE',T59,'ZPLACE',T71,'YEAR',T81,'MONTH',
     2  T92,'DAY',T99,'HOUR',T111,'MINUTE'/
     3  T35,F6.0,T47,F6.0,T59,F6.0,T71,I4,T81,I5,T92,I3,T99,I4,T111,
     4     I6//)
 4000 FORMAT(' PACKED LOCAL JULIAN DAY AND HOUR     = ',I6/
     1  ' PACKED GREENWICH JULIAN DAY AND HOUR = ',I6)
C
      END
```

```
      PROGRAM XECAMS
C
      INCLUDE MESAGE
      INCLUDE FILES
C
       DATA UAFILE,LYR2FL,TRRNFL,LYR4FL,GRIDF1,GRIDF2,GRIDF3,XOTFIL,
     1   OBFILE,XINFIL,GWCOBF,GWCFIL,SPAREF,SFCOBF,SFCFIL,MICTRF,
     2   MOTFIL/1,2,3,4,5,7,9,6,8,10,11,12,13,14,15,16,17/
C
      NAMELIST /XFLAGS/ FUNCTN, TASK, JOB, BLOCK, FILE, PRINT, LEVEL
 5    READ(XINFIL,XFLAGS, END=110)
      WRITE(XOTFIL,2000) FUNCTN, TASK, JOB, BLOCK, FILE,
     1     PRINT, LEVEL
C
C**   BRANCH ON THE VALUE OF FUNCTN
      GO TO (20,40,60,80), FUNCTN
C
 20   CALL APPLY
      GO TO 100
 40   CALL ANALYZ
      GO TO 100
 60   CONTINUE
C**   CALL OTHER ROUTINES
      GO TO 100
 80   CALL PLTPRG
C
 100  IF(JOB.GE.0) GO TO 5
 110  CONTINUE
C
 1000 FORMAT(7I3)
 2000 FORMAT('1INPUT FLAGS TO MESAGE COMMON'//T2,'FUNCTION',
     1  T15,'TASK',T26,'JOB',T34,'BLOCK',T45,'FILE',T55,'PRINT',
     2  T66,'LEVEL'/T8,I2,T17,I2,T26,I2,T37,I2,T47,I2,
     3  T58,I2,T69,I2//)
      STOP
      END
```

D.3.2 Data Analysis Routines (DATANL)

```
      COMPILER(DIAG=3)
      SUBROUTINE DATANL
C
      INCLUDE MESAGE
      INCLUDE COORD
      INCLUDE MRCIPF
      INCLUDE FILES
      INCLUDE MRCDAT
      INCLUDE INTLYZ
      INCLUDE COEF
C
      INCLUDE DATFLG
C
C**   INITIALIZE:
      DO 1 I=1,3
      NVFLAG(I) = 0
 1    CONTINUE
      WRITE(XOTFIL,10)
   10 FORMAT('1DATA ANALYSIS'//)
C
C**   CHECK WHETHER USER INITIALIZATION DATA IS AVAILABLE
C
      IF(IDAT(6).NE.0) GO TO 50
      CALL MANUAL
      RETURN
 50   CONTINUE
C
C
C
C**   CHECK WHETHER MESOMETEOROLOGICAL DATA IS AVAILABLE
C
      IF(IDAT(5).NE.0) GO TO 100
      CALL METVAR
 100  CONTINUE
C
C
      IF(NVFLAG(3).EQ.1) GO TO 200
C
C
C**   CHECK WHETHER MESOMODEL DATA IS AVAILABLE AND RECENT
      IF(IDAT(4).NE.0) GO TO 200
      IF(IDIFTM(TIME,UPTIME).GT.3) GO TO 200
      CALL MESVAR
 200  CONTINUE
C
C
C**   OBTAIN UPPER AIR PROFILES OF PRESSURE, POTENTIAL TEMPERATURE,
C**   AND WIND
      CALL PROFLS
      IF(IFLG1.EQ.0) GO TO 1000
C
C
C**   OBTAIN POTENTIAL TEMPERATURE PROFILE AT SITE
      CALL TMPPRF
C
C
```

```
C**   CHECK WHETHER SURFACE STATIONS SHOULD BE USED FOR
C**   SURFACE TEMPERATURE
      IF(NVFLAG(2).EQ.1) GO TO 400
      IF(NSFNUM.EQ.0) GO TO 1000
      CALL SRFTMP
 400  CONTINUE
C
C
C**   CHECK WHETHER UA DATA SHOULD BE USED FOR GENERAL WIND
      IF(NVFLAG(3).EQ.1) GO TO 1000
      CALL UAVAR
C
C
 1000 CONTINUE
C
C**   DATA ANALYSIS COMPLETE
C**   CHECK FLAGS TO SEE WHETHER MANUAL INPUT IS REQUIRED
C
      ITMP = NVFLAG(1)*NVFLAG(2)*NVFLAG(3)
      IF(ITMP.EQ.0) IHIERC = 0
      IF(ITMP.EQ.0) WRITE(XOTFIL,6000)
C
      WRITE(XOTFIL,5000) TSITE,TPSITE,(TPRO(J),ZPRO(J),J=1,LEVELS)
     1 ,GENWND
 5000 FORMAT(1X,'TEMPERATURE AT SITE',T34,'=',T40,F8.2,'  DEG K'//
     1  1X,'POTENTIAL TEMPERATURE AT SITE',T34,'=',T40,F8.2,'  DEG K'
     2  //1X,'TEMPERATURE PROFILE',3(T40,F8.2,10X,F8.2/)/
     3  1X,'GENERAL WIND '/T20,'U COMPONENT',T34,'=',T40,F8.2/
     4  T20, 'V COMPONENT',T34,'=',T40,F8.2)
 6000 FORMAT('1DATA ANALYSIS UNABLE TO INITIALIZE WINDEX'/
     1          ' RESTART WITH MANUAL INPUT')
C
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE GEOWND(X,Y,ZSURF,VX,VY)
C
C*****************************************************************************
C*
C*    THIS ROUTINE USES THE UPPER AIR DATA INTERPOLATION PLANES
C*    TO COMPUTE A 'SURFACE' GEOSTROPHIC WIND V AT A GIVEN LOCATION
C*    (X,Y).  V SATISFIES THE EQUATION
C*
C*                A = -2.0*OMEGA*SIN(PHI)*(K X V)
C*    WHERE
C*                A = HORIZONTAL ACCELERATION
C*            OMEGA = ROTATIONAL ANGULAR VELOCITY OF EARTH
C*              PHI = LATITUDE OF LOCATION (X,Y) (RADIANS)
C*                K = VERTICAL UNIT VECTOR
C*            K X V = CROSS PRODUCT OF K AND V
C*
C*    A IS COMPUTED FROM THE EXPRESSION
C*
C*                A = G*GRZ - G*(GRT/T)*(Z-ZSURF)
C*    WHERE
C*                G = GRAVITY
C*                Z = ELEVATION OF A CONSTANT PRESSURE SURFACE
C*                    ABOVE LOCATION (X,Y)
C*              GRZ = HORIZONTAL GRADIENT OF Z
C*                T = TEMPERATURE AT SPATIAL LOCATION (X,Y,Z)
C*              GRT = HORIZONTAL GRADIENT OF T
C*            ZSURF = ELEVATION AT LOCATION (X,Y)
C*
C*    INPUTS:
C*                X = UTM X COORDINATE OF INPUT LOCATION
C*                Y = UTM Y COORDINATE
C*             ZSURF = SEE ABOVE
C*          AA(...) = COEFFICIENTS OF
C*          BB(...) =   UPPER AIR DATA INTERPOLATION
C*          CC(...) =   PLANES (PASSED VIA /COEF/ COMMON)
C*    OUTPUTS:
C*               VX = X COMPONENT OF GEOSTROPHIC WIND
C*               VY = Y COMPONENT OF GEOSTROPHIC WIND
C*    LOCAL VARIABLES:
C*              PHI = SEE ABOVE
C*            OMEGA = SEE ABOVE
C*                G = SEE ABOVE
C*               L1 = CONSTANT PRESSURE LEVEL JUST ABOVE SURFACE
C*                T = SEE ABOVE
C*                Z = SEE ABOVE
C*             GRZX = X COMPONENT OF GRZ
C*             GRZX = Y COMPONENT OF GRZ
C*             GRTX = X COMPONENT OF GRT
C*             GRTY = Y COMPONENT OF GRT
C*               AX = X COMPONENT OF A
C*               AY = Y COMPONENT OF A
C*
C*****************************************************************************
C
```

```
      INCLUDE COEF
C
      DATA OMEGA/7.272E-5/, G/9.8/, CONVRT/1.571E-4/
C
C
C**   DETERMINE LATITUDE FROM UTM Y COORDINATE
      PHI = Y*CONVRT
C
C**   DETERMINE WHICH OF 850.700.500 MB PRESSURE SURFACES IS CLOSEST TO
C**   BUT ABOVE GROUND LEVEL
      L1 = 0
   10 L1 = L1+1
       Z = AA(L1.4) + BB(L1.4)*X + CC(L1.4)*Y
      IF(ZSURF.GT.Z.AND.L1.LT.3) GO TO 10
C
C**   COMPUTE TEMPERATURE AT (X.Y.Z)
      T = AA(L1.1) + BB(L1.1)*X + CC(L1.1)*Y
C
C**   COMPUTE GRADIENT COMPONENTS
C**   .001 OCCURS SINCE X.Y ARE MEASURED IN KILOMETERS
      GRZX = BB(L1.4)*.001
      GRZY = CC(L1.4)*.001
      GRTX = BB(L1.1)*.001
      GRTY = CC(L1.1)*.001
C
C**   COMPUTE HORIZONTAL ACCELERATION COMPONENTS
      ZDIFF = Z-ZSURF
      AX = G*(GRZX-(GRTX/T)*ZDIFF)
      AY = G*(GRZY-(GRTY/T)*ZDIFF)
C
C**   COMPUTE VELOCITY COMPONENTS
      FACTOR = 1.0/(2.0*OMEGA*SIN(PHI))
      VX = -AY*FACTOR
      VY = AX*FACTOR
C
      RETURN
      END
```

```
      SUBROUTINE INTCOF(X,Y,N,A,B,C,NCFLAG)
C
C*****************************************************************************
C*
C*    ROUTINE INTCOF (INTERPOLATION COEFFICIENTS) CALCULATES
C*    COEFFICIENTS A,B,C OF FUNCTIONS F(X,Y) = A+BX+CY USED FOR
C*    INTERPOLATING UPPER AIR DATA.  N LOCATIONS ARE GIVEN AS INPUT;
C*    N FUNCTIONS ARE RETURNED AS OUTPUT.
C*
C*    **VARIABLES**
C*
C*      INPUTS:
C*              X(.) = X,Y COORDINATES OF
C*              Y(.) =   INPUT LOCATIONS
C*                 N = NUMBER OF INPUT LOCATIONS
C*      OUTPUTS:
C*              A(.) = COEFFICIENTS OF OUTPUT FUNCTIONS
C*              B(.) = COEFFICIENTS OF OUTPUT FUNCTIONS
C*              C(.) = COEFFICIENTS OF OUTPUT FUNCTIONS
C*           NCFLAG = SUCCESS FLAG
C*
C*****************************************************************************
C
      PARAMETER NDIM=3
      DIMENSION X(N), Y(N), A(N), B(N), C(N)
      DIMENSION R(NDIM,NDIM), RI(NDIM,NDIM), LL(NDIM), KK(NDIM)
C
C
      DO 10 I=1,NDIM
      DO 10 J=1,NDIM
 10   R(I,J) = 0.0
C
      DO 20 I=1,N
      R(1,2) = R(1,2)+X(I)
      R(1,3) = R(1,3)+Y(I)
      R(2,2) = R(2,2)+X(I)**2
      R(2,3) = R(2,3)+X(I)*Y(I)
      R(3,3) = R(3,3)+Y(I)**2
 20   CONTINUE
C
C**   R IS A SYMMETRIC MATRIX
C
      R(1,1) = N
      R(2,1) = R(1,2)
      R(3,1) = R(1,3)
      R(3,2) = R(2,3)
C
      DO 30 I=1,3
      DO 30 J=1,3
 30   RI(I,J) = R(I,J)
```

```
      CALL MINV(RI,NDIM,DET,KK,LL)
      IF(ABS(DET).LT.1.0E-05) GO TO 1000
      NCFLAG = 1
C
      DO 500 I=1,N
      A(I) = RI(1,1)+RI(1,2)*X(I)+RI(1,3)*Y(I)
      B(I) = RI(2,1)+RI(2,2)*X(I)+RI(2,3)*Y(I)
      C(I) = RI(3,1)+RI(3,2)*X(I)+RI(3,3)*Y(I)
 500  CONTINUE
      RETURN
C
 1000 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE INTERP(X,Y,N,X0,Y0)
C
C*****************************************************************************
C*
C*    ROUTINE INTERP (INTERPOLATION) RETURNS A VALUE YO INTERPOLATED
C*    FROM A TABLE OF N VALUES X(.),Y(.) USING AN ARGUMENT XO.  THE
C*    TABLE ARGUMENTS X(.) MUST BE MONOTONIC (INCREASING OR DECREASING)
C*
C*****************************************************************************
C
      DIMENSION X(N), Y(N)
C
C**   DETERMINE 2 TABLE ARGUMENTS CLOSEST TO XO
      N1 = N-1
C**   X(.) IS DECREASING
C**   IF X(.) IS DECREASING, BRANCH TO 25
      IF(X(1).GT.X(2)) GO TO 25
C**   X(.) IS INCREASING
      DO 10 I=2,N1
      JJ = I
      IF(X0.LE.X(JJ)) GO TO 50
 10   CONTINUE
      JJ = N
      GO TO 50
C
 25   CONTINUE
      DO 30 I=2,N1
      JJ = I
      IF(X0.GE.X(JJ)) GO TO 50
 30   CONTINUE
      JJ = N
C
 50   J = JJ-1
C**   THE TWO ARGUMENTS CLOSEST TO XO ARE X(J) AND X(JJ)
      Y0 = ((Y(JJ)-Y(J))/(X(JJ)-X(J)))*(X0-X(J))+Y(J)
C
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE MANUAL
C
      INCLUDE MESAGE
      INCLUDE FILES
      INCLUDE INTLYZ
      INCLUDE COORD
      DIMENSION PLEVLS(3)
      DATA PLEVLS/85000.0,70000.0,50000.0/
      DATA P2/.2862/
C
C
C**   READ IN MESOSCALE DRIVERS AND SPOT OBSERVATIONS
      NAMELIST /MFLAGS/ DELQ,TPRO,ZPRO,GENWND,NUMOBS
      READ(XINFIL,MFLAGS)
      IF(NUMOBS.EQ.0) GO TO 25
      READ(XINFIL,15) (KINDX(I),LINDX(I),THETA0(I),UW(I),VW(I),I=1
     1                ,NUMOBS)
   25 CONTINUE
C
C**   ESTIMATE ACTUAL SURFACE TEMPERATURE
      ZZ = ZPLACE
      CALL INTERP(ZPRO,PLEVLS,LEVEL,ZZ,PP)
      CALL INTERP(ZPRO,TPRO,LEVEL,ZZ,TPEX)
      TPSITE = TPEX + DELQ*(100000./PP)**P2
      TSITE = TPSITE*(PP/100000.0)**P2
C
      IF(NUMOBS.EQ.0) RETURN
C
      WRITE(XOTFIL,35) (KINDX(I),LINDX(I),THETA0(I),UW(I),VW(I),I=1,
     1     NUMOBS)
   35 FORMAT(1X,'SURFACE OBSERVATIONS'//1X,'GRID INDICES',T24,
     1  'POTENTIAL TEMPERATURE',T55,'U WIND',T71,'V WIND'//
     2  (T2,I5,T9,I5,T35,F10.2,T53,F8.2,T69,F8.2//))
   15 FORMAT(2I5,3F10.2)
C
      RETURN
      END
```

```
      SUBROUTINE MESVAR
C
      PARAMETER NVAR=4. LEVELS=3
      INCLUDE MESAGE '
      INCLUDE FILES
      INCLUDE MRCDAT
      INCLUDE INTLYZ
      INCLUDE DATFLG
      DATA PI,Z0,LM,RHO,ALOGF/3.14159,.1,2,1.12,6.908/
C
C
C**   ESTIMATE GENERAL WIND USING MESOMODEL )ATA
      I = 1
C
      H = HMM(LM,I)-ZGRDMM(I)
      RMAG = AVM(LM,I)**2+AVN(LM,I)**2
      AS = 4*(AVM1(LM,I)**2+AVN1(LM,I)**2)/RMAG
C
      F1 = ALOG(H/Z0)
      F2 = 9.0*((F1-1)**2-AS*(F1-.5)**2)/(AS-2.25)
      IF(F2.LT.0.0) F2=0.0
      AL = SQRT(F2)
C
      RHUK = SQRT(RMAG/((F1-1)**2+(AL/3)**2))
C
      C11 = 2*AVM1(LM,I)-AVM(LM,I)
      C12 = 2*AVN1(LM,I)-AVN(LM,I)
      C21 = 4*AVN1(LM,I)-3*AVN(LM,I)
      C22 = -4*AVM1(LM,I)+3*AVM(LM,I)
C
      D = C11*C22-C21*C12
      IF(D.EQ.0.0) GO TO 4805
      F3 = RHUK/2.0
      COSA = F3*C22/D
      SINA = -F3*C21/D
      GO TO 4810
 4805 DENOM = SQRT(RMAG)
      COSA = AVM(LM,I)/DENOM
      SINA = AVN(LM,I)/DENOM
 4810 CONTINUE
      UK = RHUK/(H*RHO)
      GENWND(1) = UK*ALOGF*COSA
      GENWND(2) = UK*ALOGF*SINA
C
C
      WRITE(XOTFIL,5000)
 5000 FORMAT(' SURFACE WIND GIVEN BY MESOMODEL DATA ANALYSIS'//)
C
      NVFLAG(3) = 1
      RETURN
      END
```

```
C**      SUBROUTINE MINV
C
C**      PURPOSE
C**         INVERT A MATRIX
C
C**      USAGE
C**         CALL MINV(A,N,D,L,M)
C
C**      DESCRIPTION OF PARAMETERS
C**         A - INPUT MATRIX. DESTROYED IN COMPUTATION AND REPLACED BY
C**             RESULTANT INVERSE.
C**         N - ORDER OF MATRIX A
C**         D - RESULTANT DETERMINANT
C**         L - WORK VECTOR OF LENGTH N
C**         M - WORK VECTOR OF LENGTH N
C
C**      REMARKS
C**         MATRIX A MUST BE A GENERAL MATRIX
C
C**      SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C**         NONE
C
C**      METHOD
C**         THE STANDARD GAUSS-JORDAN METHOD IS USED. THE DETERMINANT
C**         IS ALSO CALCULATED. A DETERMINANT OF ZERO INDICATES THAT
C**         THE MATRIX IS SINGULAR.
C
C**   ..................................................................
C
      SUBROUTINE MINV(A,N,D,L,M)
      DIMENSION A(1),L(1),M(1)
C
C**      ..................................................................
C
C**      IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE
C**      C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION
C**      STATEMENT WHICH FOLLOWS.
C
C**   DOUBLE PRECISION A,D,BIGA,HOLD
C
C**      THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS
C**      APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS
C**      ROUTINE.
C
C**      THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO
C**      CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  ABS IN STATEMENT
C**      10 MUST BE CHANGED TO DABS.
C
C**      ..................................................................
C
```

```
C**      SEARCH FOR LARGEST ELEMENT
C
      D=1.0
      NK=-N
      DO 80 K=1,N
      NK=NK+N
      L(K)=K
      M(K)=K
      KK=NK+K
      BIGA=A(KK)
      DO 20 J=K,N
      IZ=N*(J-1)
      DO 20 I=K,N
      IJ=IZ+I
   10 IF( ABS(BIGA)- ABS(A(IJ))) 15,20,20
   15 BIGA=A(IJ)
      L(K)=I
      M(K)=J
   20 CONTINUE
C
C**      INTERCHANGE ROWS
C
      J=L(K)
      IF(J-K) 35,35,25
   25 KI=K-N
      DO 30 I=1,N
      KI=KI+N
      HOLD=-A(KI)
      JI=KI-K+J
      A(KI)=A(JI)
   30 A(JI) =HOLD
C
C**      INTERCHANGE COLUMNS
C
   35 I=M(K)
      IF(I-K) 45,45,38
   38 JP=N*(I-1)
      DO 40 J=1,N
      JK=NK+J
      JI=JP+J
      HOLD=-A(JK)
      A(JK)=A(JI)
   40 A(JI) =HOLD
C
C**      DIVIDE COLUMN BY MINUS PIVOT (VALUE OF PIVOT ELEMENT IS
C**      CONTAINED IN BIGA)
C
   45 IF(BIGA) 48,46,48
   46 D=0.0
      RETURN
```

```
   48 DO 55 I=1,N
      IF(I-K) 50,55,50
   50 IK=NK+I
      A(IK)=A(IK)/(-BIGA)
   55 CONTINUE
C
C**      REDUCE MATRIX
C
      DO 65 I=1,N
      IK=NK+I
      HOLD=A(IK)
      IJ=I-N
      DO 65 J=1,N
      IJ=IJ+N
      IF(I-K) 60,65,60
   60 IF(J-K) 62,65,62
   62 KJ=IJ-I+K
      A(IJ)=HOLD*A(KJ)+A(IJ)
   65 CONTINUE
C
C**      DIVIDE ROW BY PIVOT
C
      KJ=K-N
      DO 75 J=1,N
      KJ=KJ+N
      IF(J-K) 70,75,70
   70 A(KJ)=A(KJ)/BIGA
   75 CONTINUE
C
C**      PRODUCT OF PIVOTS
C
      D=D*BIGA
C
C**      REPLACE PIVOT BY RECIPROCAL
C
      A(KK)=1.0/BIGA
   80 CONTINUE
C
C**      FINAL ROW AND COLUMN INTERCHANGE
C
      K=N
  100 K=(K-1)
      IF(K) 150,150,105
  105 I=L(K)
      IF(I-K) 120,120,108
  108 JQ=N*(K-1)
      JR=N*(I-1)
      DO 110 J=1,N
      JK=JQ+J
      HOLD=A(JK)
      JI=JR+J
      A(JK)=-A(JI)
  110 A(JI) =HOLD
```

```
120 J=M(K)
    IF(J-K) 100,100,125
125 KI=K-N
    DO 130 I=1,N
    KI=KI+N
    HOLD=A(KI)
    JI=KI-K+J
    A(KI)=-A(JI)
130 A(JI) =HOLD
    GO TO 100
150 RETURN
    END
```

```
    SUBROUTINE NETVAR
    RETURN
    END
```

```
      SUBROUTINE PROFLS
C
C**   THIS ROUTINE OBTAINS FUNCTIONS WHICH INTERPOLATE IN SPACE AND
C**   UPDATE IN TIME THE UPPER AIR PROFILES OF T, U, V, AND Z FOR THE
C**   850, 700, AND 500 MILLIBAR PRESSURE LEVELS
C
C
      INCLUDE MESAGE
      INCLUDE FILES
      INCLUDE COORD
      INCLUDE MRCIPF
      INCLUDE MRCDAT
      INCLUDE COEF
      INCLUDE DATFLG
      DIMENSION UADATA(LEVELS,NUAX,NVAR)
      DIMENSION GWDATA(4,ITT,NGWX,NVAR)
      EQUIVALENCE (GWDATA(1,1,1,1),TGW(1,1,1))
C
      DIMENSION UACORS(NLVL,NUAX,8)
      EQUIVALENCE (UACORS(1,1,1), PRES(1,1))
C
      DIMENSION A(NGWX), B(NGWX), C(NGWX)
      DIMENSION AGW(LEVELS,ITT,NVAR)
      DIMENSION BGW(LEVELS,ITT,NVAR)
      DIMENSION CGW(LEVELS,ITT,NVAR)
C
      DIMENSION X(NLVL),Y(NLVL)
      DIMENSION XLOC(NGWX), YLOC(NGWX)
      DIMENSION PLEVLS(LEVELS)
      DIMENSION IUAVAR(NVAR)
C
      DATA PLEVLS/85000.0, 70000.0, 50000.0/
      DATA IUAVAR /6,7,8,4/
      DATA P2/.2862/
C
C**   INITIALIZE:
C
      NCFLAG = 0
      IFLG1 = 0
      ISKIP = 0
      NG1 = NGWNUM(1)
      NG2 = NGWNUM(2)
      DO 10 ITR=1,2
      DO 10 IVAR=1,NVAR
      DO 10 L=1,LEVELS
      AGW(L,ITR,IVAR) = 0.0
      BGW(L,ITR,IVAR) = 0.0
      CGW(L,ITR,IVAR) = 0.0
   10 CONTINUE
C
C**   CHECK GWC DATA
C**   NG2.LT.3 MEANS NO GWC PLANES ARE AVAILABLE FOR UPDATING
C**   SEE IF UA DATA IS RECENT
      IF(NG2.LT.3) GO TO 800
      IFLG1 = NG2
C
```

```
C**   GENERATE SPATIAL INTERPOLATION FUNCTIONS FOR GWC PROFILES AT
C**   SIMULATION TIME
      ITR = 2
  100 CONTINUE
      M1 = NGWNUM(ITR)
      DO 105 J=1,M1
      XLOC(J) = XGRD(ITR,J)
  105 YLOC(J) = YGRD(ITR,J)
C
C
C**   OBTAIN BASIS FUNCTIONS
      CALL INTCOF(XLOC,YLOC,NGWNUM(ITR),A,B,C,NCFLAG)
      IF(NCFLAG.EQ.-1) GO TO 800
C
C**   GENERATE DATA INTERPOLATION FUNCTIONS
      DO 115 L=1,LEVELS
      LL = L+1
      DO 115 ISTN=1,M1
C**   TEMPERATURE IS CONVERTED TO POTENTIAL TEMPERATURE
      RF = GWDATA(LL,ITR,ISTN,1)*(100000.0/PGW(LL,ITR,ISTN))**P2
      AGW(L,ITR,1) = AGW(L,ITR,1)+RF*A(ISTN)
      BGW(L,ITR,1) = BGW(L,ITR,1)+RF*B(ISTN)
      CGW(L,ITR,1) = CGW(L,ITR,1)+RF*C(ISTN)
  115 CONTINUE
C
      DO 120 IVAR=2,4
      DO 120 L=1,LEVELS
      LL = L+1
      DO 120 ISTN=1,M1
      AGW(L,ITR,IVAR) = AGW(L,ITR,IVAR)+GWDATA(LL,ITR,ISTN,IVAR)*A(ISTN)
      BGW(L,ITR,IVAR) = BGW(L,ITR,IVAR)+GWDATA(LL,ITR,ISTN,IVAR)*B(ISTN)
      CGW(L,ITR,IVAR) = CGW(L,ITR,IVAR)+GWDATA(LL,ITR,ISTN,IVAR)*C(ISTN)
  120 CONTINUE
C
C**   INTERPOLATION FUNCTIONS:  AA(L,I)+BB(L,I)*X+CC(L,I)*Y
C**   I=1:  POTEMP     I=2:  UWIND     I=3:  VWIND     I=4:  ELEVATION
C**   L=1:  850 LEVEL     L=2:  700 LEVEL     L=3:  500 LEVEL
C
      IF(NUANUM.EQ.0) GO TO 200
C**   ITR=1 MEANS PLANES HAVE BEEN OBTAINED AT BOTH UA AND
C**   SIMULATION TIME
      IF(ITR.EQ.1) GO TO 125
C**   ITR=2 MEANS THE SIMULATION-TIME PASS HAS JUST BEEN COMPLETED
C**   CHECK TO SEE IF THERE ARE AT LEAST 3 GWC STATIONS FOR UA TIME
      IF(NG1.LT.3) GO TO 800
      ITR = 1
      GO TO 100
  125 CONTINUE
C
```

```
C**   ACQUIRE UA DATA AT 850. 700. 500 LEVELS
      DO 195 ISTN=1.NUANUM
C**   LOAD UPPER AIR PRESSURE PROFILE
      DO 140 I=1.NLVL
 140  X(I) = PRES(I.ISTN)
C
C**   LOOP OVER VARIABLE TYPES AND ISOBARIC PRESSURE LEVELS
      DO 190 IVAR=1.NVAR
      JVAR = IUAVAR(IVAR)
      DO 180 L=1.LEVELS
C
C**   INTERPOLATE FROM PROFILES TO GET 850.700.500 MILIBAR
C**   TEMPERATURES. WINDS. AND ELEVATIONS
      DO 160 I=1.NLVL
 160  Y(I) = UACORS(I.ISTN.JVAR)
      CALL INTERP(X.Y.NTR(ISTN).PLEVLS(L).UADATA(L.ISTN.IVAR))
C
C**   UPDATE UA DATA WITH GWC DATA
      IF(ISKIP.EQ.1) GO TO 180
      UADATA(L.ISTN.IVAR) = UADATA(L.ISTN.IVAR)+AGW(L.2.IVAR)
     M  -AGW(L.1.IVAR)+(BGW(L.2.IVAR)-BGW(L.1.IVAR))*XSTA(ISTN)
     M                 +(CGW(L.2.IVAR)-CGW(L.1.IVAR))*YSTA(ISTN)
 180  CONTINUE
C
 190  CONTINUE
 195  CONTINUE
C
C**   GENERATE FUNCTIONS WHICH INTERPOLATE UPDATED UA DATA:
C**   BRANCH ON THE NUMBER OF UA STATIONS
      IF(NUANUM.GE.3) GO TO 500
      GO TO (300.400). NUANUM
C
 200  CONTINUE
      DO 210 L=1.LEVELS
      DO 210 IVAR=1.NVAR
      AA(L.IVAR) = AGW(L.2.IVAR)
      BB(L.IVAR) = BGW(L.2.IVAR)
 210  CC(L.IVAR) = CGW(L.2.IVAR)
      GO TO 1000
C
 300  CONTINUE
      DO 310 L=1.LEVELS
      DO 310 IVAR=1.NVAR
      AA(L.IVAR) = UADATA(L.1.IVAR)-BGW(L.2.IVAR)*XSTA(1)-
     M  CGW(L.2.IVAR)*YSTA(1)
      BB(L.IVAR) = BGW(L.2.IVAR)
 310  CC(L.IVAR) = CGW(L.2.IVAR)
      GO TO 1000
```

```
C
 400  CONTINUE
      DELX = XSTA(2)-XSTA(1)
      DELY = YSTA(2)-YSTA(1)
      DST =  DELX**2 + DELY**2
      DO 410 L=1,LEVELS
      DO 410 IVAR=1,NVAR
      DELZ = UADATA(L,2,IVAR)-UADATA(L,1,IVAR)
      GRADTA = CGW(L,2,IVAR)*DELX-BGW(L,2,IVAR)*DELY
      BB(L,IVAR) = (DELZ*DELX-GRADTA*DELY)/DST
      CC(L,IVAR) = (DELZ*DELY+GRADTA*DELX)/DST
 410  AA(L,IVAR) = UADATA(L,2,IVAR)-BB(L,IVAR)*XSTA(2)-CC(L,IVAR)*YSTA(
     M  2)
      GO TO 1000
C
 500  CONTINUE
      CALL INTCOF(XSTA,YSTA,NUANUM,A,B,C,NCFLAG)
      IF(NCFLAG.EQ.-1) IFLG1=2
      IF(IFLG1.EQ.2) GO TO 400
      DO 510 IVAR=1,NVAR
      DO 510 L=1,LEVELS
      AA(L,IVAR) = 0.0
      BB(L,IVAR) = 0.0
 510  CC(L,IVAR) = 0.0
      DO 515 IVAR=1,NVAR
      DO 515 L=1,LEVELS
      DO 515 ISTN=1,NUANUM
      AA(L,IVAR) = AA(L,IVAR)+UADATA(L,ISTN,IVAR)*A(ISTN)
      BB(L,IVAR) = BB(L,IVAR)+UADATA(L,ISTN,IVAR)*B(ISTN)
 515  CC(L,IVAR) = CC(L,IVAR)+UADATA(L,ISTN,IVAR)*C(ISTN)
      GO TO 1000
C
C**   THIS IS A TROUBLE POINT:
C**   PROGRAM FLOW HERE INDICATES SOME
C**   SHORTCOMING IN THE DATA
 800  CONTINUE
      ISKIP = 1
      IDIFF = IDIFTM(TIME,UPTIME(1))
C**   IF UA DATA IS OLD AND GWC INFO AT SIMULATION TIME IS
C**   INCOMPLETE, PUNT
      IF(IDIFF.GT.3.AND.NG2.LT.3) RETURN
C**   IF UA DATA IS OLD AND CAN'T BE UPDATED, USE GWC
C**   SIMULATION-TIME PLANES
      IF(IDIFF.GT.3) GO TO 200
C**   IF UADATA IS RECENT, USE IT (AND POSSIBLY GWC INFO)
C**   TO GET PLANES
      IFLG1 = MAX0(NUANUM,NG2)
      IF(NCFLAG.EQ.-1) IFLG1=NUANUM
      GO TO 125
C
 1000 CONTINUE
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE SRFTMP
C
      INCLUDE MESAGE
      INCLUDE COORD
      INCLUDE MRCIPF
      INCLUDE FILES
      INCLUDE MRCDAT
      INCLUDE COEF
      INCLUDE INTLYZ
      INCLUDE DATFLG
C
      DIMENSION PLEVLS(LEVELS), T(LEVELS), Z(LEVELS)
      DATA PLEVLS/85000.0,70000.0,50000.0/
      DATA P2/.2862/
C
C
C
C**   USE PROFILES AND SURFACE STATION DATA TO ESTIMATE
C**        POTENTIAL TEMPERATURE AT SITE
C
      SUMD = 0.0
      DO 200 I=1,NSFNUM
C
      DO 100 L=1,LEVELS
      T(L) = AA(L,1)+BB(L,1)*XSTS(I)+CC(L,1)*YSTS(I)
      Z(L) = AA(L,4)+BB(L,4)*XSTS(I)+CC(L,4)*YSTS(I)
  100 CONTINUE
C
C**   PROJECT POTENTIAL TEMPERATURE TO SURFACE AT STATION SITE
C
      CALL INTERP(Z,T,LEVELS,ZSTS(I),TT)
C
C**   CONVERT ACTUAL SURFACE TEMPERATURE TO POTENTIAL TEMP
C
      CALL INTERP(Z,PLEVLS,LEVELS,ZSTS(I),PP)
      TPSURF = TSSF(I)*(100000.0/PP)**P2
C
C**   ACCUMULATE SUM FOR AVERAGE SURFACE HEATING
C
      SUMD = SUMD+(TPSURF-TT)*TSSF(I)/TT
  200 CONTINUE
C
      DELQ = SUMD/NSFNUM
C
C**   PROJECT POTENTIAL TEMPERATURE TO SURFACE AT SITE AND
C**        APPLY CORRECTION
C**   CALCULATE TSITE FOR LATER USE
C
      ZZ = ZPLACE
      CALL INTERP(ZPRO,PLEVLS,LEVELS,ZZ,PP)
      CALL INTERP(ZPRO,TPRO,LEVELS,ZZ,TPEX)
      TPSITE = TPEX + DELQ*(100000./PP)**P2
      TSITE = TPSITE*(PP/100000.0)**P2
C
      WRITE(XOTFIL,2000)
 2000 FORMAT(' SURFACE TEMPERATURE GIVEN BY SURFACE STATIONS'//)
C
      NVFLAG(2) = 1
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE UAVAR
C
      INCLUDE MESAGE
      INCLUDE COORD
      INCLUDE MRCIPF
      INCLUDE FILES
      INCLUDE MRCDAT
      INCLUDE INTLYZ
      INCLUDE COEF
      INCLUDE DATFLG
C
      DIMENSION SUM(4)
      DIMENSION PLEVLS(3)
      DATA PLEVLS/85000.0,70000.0,50000.0/
      DATA PI/3.14159/
C
C
C
C**   IF NO UA PROFILES EXIST,USE CLOSEST SURFACE
C**   STATION DEFAULT
      IF(IFLG1.LT.2) GO TO 8000
C**   IF LESS THAN TWO  STATIONS EXIST, USE CLOSEST SURFACE
C**   SURFACE STATION DEFAULT
      IF(NSFNUM.EQ.0) GO TO 8000
C**   ESTIMATE GENERAL WIND USING UA PROFILES AND SURFACE  STATION DATA
C
      DO 100 I=1,4
  100 SUM(I) = 0.0
C
      DO 200 I=1,NSFNUM
C
C**   COMPUTE GEOSTROPHIC WIND AT EACH SURFACE STATION
      CALL GEOWND(XSTS(I),YSTS(I),ZSTS(I),VX,VY)
      WRITE(XOTFIL,999) VX,VY
C
C**   CONVERT VX,VY TO SPEED AND DIRECTION
      SPEED = SQRT(VX**2+VY**2)
      IF(SPEED.EQ.0) GO TO 8000
      IF(VX.EQ.0.0) VX=VY*(.0001)
      THETA = ATAN(VY/VX)*180.0/PI
      IF(VX.LT.0.0) THETA=180.0+THETA
      THETA = AMOD(270.0-THETA,360.0)
      WRITE(XOTFIL,999) WDSF(I),THETA,WSSF(I),SPEED
C
C**   ACCUMULATE SUMS
      TMP = WDSF(I) - THETA
      SUM(1) = SUM(1)+TMP
      SUM(2) = SUM(2)+TMP**2
C
      TMP = (WSSF(I)-SPEED)/SPEED
      SUM(3) = SUM(3)+TMP
      SUM(4) = SUM(4)+TMP*TMP
  200 CONTINUE
C
```

```
C**   COMPUTE CORRECTIONS AND RELATIVE DEVIATIONS OF CORRECTIONS
      RSDD = 0.0
      RSDS = 0.0
      DCOR = SUM(1)/NSFNUM
      RSDD = SQRT((SUM(2)-NSFNUM*DCOR**2)/(NSFNUM-1))/ABS(DCOR)
      SCORP = SUM(3)/NSFNUM
      RSDS = SQRT((SUM(4)-NSFNUM*SCORP**2)/(NSFNUM-1))/ABS(SCORP)
      WRITE(XOTFIL,999) DCOR,SCORP,RSDD,RSDS
C
C**   COMPUTE GEOSTROPHIC WIND AT SIMULATION SITE
      XX = XPLACE
      YY = YPLACE
      ZZ = ZPLACE
      CALL GEOWND(XX,YY,ZZ,VX,VY)
C**   CONVERT TO SPEED AND DIRECTION
      SPEED = SQRT(VX**2+VY**2)
      IF(VX.EQ.0.0) VX=VY*(.0001)
      IF(VX.EQ.0.0) GO TO 8000
      THETA = ATAN(VY/VX)*180.0/PI
      IF(VX.LT.0.0) THETA=180.0+THETA
C
      THETA = AMOD(270.0-THETA,360.0)
      WRITE(XOTFIL,999) THETA,SPEED
C
C**   DECIDE WHETHER TO APPLY CORRECTIONS OR USE CLOSEST STATION READINGS
      IF(RSDD.LT..5) GO TO 510
      THETA = WDSF(1)
      WRITE(XOTFIL,5000)
      GO TO  520
  510 THETA = THETA+DCOR
      WRITE(XOTFIL,5500)
  520 CONTINUE
C
      IF(RSDS.LT..5) GO TO  530
      SPEED = WSSF(1)
      WRITE(XOTFIL,6000)
      GO TO  540
  530 SPEED = SPEED*(1+SCORP)
      WRITE(XOTFIL,6500)
C
  540 CONTINUE
C
C**   RECONVERT TO XY COORDINATES
C
      THETA = (270-THETA)*PI/180.0
      GENWND(1) = SPEED*COS(THETA)
      GENWND(2) = SPEED*SIN(THETA)
      NVFLAG(3) = 1
      RETURN
```

```
C
 3000 CONTINUE
      WRITE(XOTFIL,8050)
      THETA = WDSF(1)
      SPEED = WSSF(1)
      GO TO  540
C
C
  999 FORMAT( )
 5000 FORMAT(' WIND DIRECTION GIVEN BY CLOSEST SURFACE STATION'//)
 5500 FORMAT(' WIND DIRECTION GIVEN BY GEOSTROPHIC MODEL'//)
 6000 FORMAT(' WIND SPEED GIVEN BY CLOSEST SURFACE STATION'//)
 6500 FORMAT(' WIND SPEED GIVEN BY GEOSTROPHIC MODEL'//)
 8050 FORMAT(' WINDS GIVEN BY CLOSEST SURFACE STATION'//)
      END
```

D.3.3 Surface Wind Analysis Routines (WINDEX)

```
      SUBROUTINE ACCSQL(IPT,ASQL,ACNORM)
C
C*****************************************************************
C*
C*    ACCSQL ESTIMATES THE INTEGRAL OF THE RESIDUAL ACCELERATION SQUARED
C*    OVER A LOCAL VOLUME ELEMENT.
C*
C*    INPUTS:
C*    IPT               -   INDEX SPECIFYING LOCAL BOX TO BE USED
C*    HD1(.)            -   FROM /LOCAL/
C*    HD2(.)            -   FROM /LOCAL/
C*    AR(.)             -   FROM /LOCAL/
C*    ER(.)             -   FROM /LOCAL/
C*    BUOY(.)           -   FROM /LOCAL/
C*    UL(.)             -   FROM /LOCAL/
C*    VL(.)             -   FROM /LOCAL/
C*    DD                -    FROM /INPPRM/
C*    OUTPUTS:
C*    ASQL              -    ESTIMATE OF ACCELERATION SQUARE RESIDUAL
C*                           OVER LOCAL VOLUME ELEMENT
C*    ACNRML            -    NORMAL COMPONENT OF ACCELERATION
C*    IMPORTANT LOCAL VARIABLES:
C*    W                 -    SURFACE NORMAL VELOCITY (UP TO FACTOR)
C*                           COMPUTED FROM MASS CONSERVATION REQUIREMENT
C*
C*****************************************************************
C
      INCLUDE LOCAL
      INCLUDE DRIVER
      INCLUDE INPPRM
C
      DIMENSION IPOINT(4,4)
      DATA IPOINT /1,2,9,8,2,3,4,9,9,4,5,6,8,9,6,7/
C
C**   INITIALIZE:
      I1 = IPOINT(1,IPT)
      I2 = IPOINT(2,IPT)
      I3 = IPOINT(3,IPT)
      I4 = IPOINT(4,IPT)
C
C**   COMPUTE BOX AVERAGE WINDS, SLOPES, AREA FACTOR,
C**   BUOYANCY, AND PROFILE
      UA = (UL(I1)+UL(I2)+UL(I4)+UL(I3))/4.0
      VA = (VL(I1)+VL(I2)+VL(I4)+VL(I3))/4.0
      SL1 = (HD1(I1)+HD1(I2)+HD1(I3)+HD1(I4))/4.0
      SL2 = (HD2(I1)+HD2(I2)+HD2(I3)+HD2(I4))/4.0
      ARAV = SQRT(1.+SL1*SL1+SL2*SL2)
      BAV = (BUOY(I1)+BUOY(I2)+BUOY(I4)+BUOY(I3))/4.0
      ERAV = (ER(I1)+ER(I2)+ER(I4)+ER(I3))/4.0
      W = AR(I3)*UL(I3)+AR(I4)*VL(I4)
     1   -AR(I1)*UL(I1)-AR(I2)*VL(I2)
      W = W/ERAV
C
```

```
C**   COMPUTE CARTESIAN COMPONENTS OF RESIDUAL ACCELERATION
C
      A1 =   UL(I1)*UL(I1)*AR(I1)/ER(I1)+UL(I2)*VL(I2)*AR(I2)/ER(I2)
     1      -UL(I3)*UL(I3)*AR(I3)/ER(I3)-UL(I4)*VL(I4)*AR(I4)/ER(I4)
     2      +W*UA
C
      A2 =   VL(I1)*UL(I1)*AR(I1)/ER(I1)+VL(I2)*VL(I2)*AR(I2)/ER(I2)
     1      -VL(I3)*UL(I3)*AR(I3)/ER(I3)-VL(I4)*VL(I4)*AR(I4)/ER(I4)
     2      +W*VA
C
      A3 =   (UL(I1)*UL(I1)*HD1(I1)+VL(I1)*UL(I1)*HD2(I1))*AR(I1)/ER(I1)
     1      +(UL(I2)*VL(I2)*HD1(I2)+VL(I2)*VL(I2)*HD2(I2))*AR(I2)/ER(I2)
     2      -(UL(I3)*UL(I3)*HD1(I3)+VL(I3)*UL(I3)*HD2(I3))*AR(I3)/ER(I3)
     3      -(UL(I4)*VL(I4)*HD1(I4)+VL(I4)*VL(I4)*HD2(I4))*AR(I4)/ER(I4)
     4      + W*(UA*SL1+VA*SL2)
C
C ***********************************************
C
C**   NORMAL COMPONENT OF ACCELERATION
      FTR = (-SL1*A1-SL2*A2+A3)/ARAV**2
      ACNORM = FTR/DD
C
C**   REMOVE NORMAL COMPONENT
      A1 = A1+SL1*FTR
      A2 = A2+SL2*FTR
      A3 = A3-FTR
C**   ADD SURFACE PARALLEL COMPONENTS OF BUOYANCY
      A1 = A1 + DD*BAV*ARAV*SL1
      A2 = A2 + DD*BAV*ARAV*SL2
      A3 = A3 + DD*BAV*(APAV-1.0)/ARAV
C ***********************************************
C
C**   LOCAL RESIDUAL
      ASQL = A1*A1+A2*A2+A3*A3
      ASQL = ASQL + FTR*FTR
      ASQL = ASQL/ARAV
C
      RETURN
      END
```

```
      SUBROUTINE ACCSQR(V1,V2,ACSQ,ACNORM)
C
C*****************************************************************************
C*
C*    ACCSQR CALLS ACCSQL TO OBTAIN ESTIMATES OF THE LOCAL ACCELERATION
C*    RESIDUAL AND NORMAL COMPONENT OF ACCELERATION.  LOCAL ESTIMATES
C*    FROM UPSTREAM BOXES ARE AVERAGED AND RETURNED TO RELAX.
C*
C*    INPUTS:
C*    V1                -    U-COMPONENT OF WIND AT LOCAL POINT 9
C*    V2                -    V-COMPONENT OF WIND AT LOCAL POINT 9
C*    NSET(.)           -    FROM / SETBOX/
C*    OUTPUTS:
C*    UL(9)             -   IN /LOCAL/
C*    VL(9)             -   IN /LOCAL/
C*    ACSQ              -   LOCAL CONTRIBUTION TO ACCELERATION RESIDUAL
C*    ACNORM            -   NORMAL COMPONENT OF ACCELERATION
C*
C*****************************************************************************
C
      INCLUDE LOCAL
      INCLUDE INPPRM
      INCLUDE SETBOX
C
C**   INITIALIZE:
      UL(9) = V1
      VL(9) = V2
      ACSQ = 0.0
      ACNORM = 0.0
C
      DO 100 I=1,4
      IF(NSET(I).EQ.0) GO TO 100
      CALL ACCSQL(I,ASQL,ACNRML)
      ACSQ = ACSQ + ASQL
      ACNORM = ACNORM + ACNRML
  100 CONTINUE
C
C**   COMPUTE AVERAGE OF LOCAL RESIDUES
      ACSQ = ACSQ*BOXAV
      ACNORM = ACNORM*BOXAV
      RETURN
      END
```

```
      SUBROUTINE DELTA(V1,V2,T,FLDVSQ,ACSQ,DTT,DUT,DVT)
C
C*****************************************************************************
C*
C*    DELTA ESTIMATES PARTIAL DERIVATIVES OF THE ACCELERATION RESIDUAL
C*    WITH RESPECT TO VELOCITY COMPONENTS AND PARTIAL DERIVATIVES OF THE
C*    TEMPERATURE RESIDUAL WITH RESPECT TO LOCAL TEMPERATURES.
C*
C*    INPUTS:
C*    V1                -   U-COMPONENT OF WIND AT LOCAL POINT 9
C*    V2                -   V-COMPONENT OF WIND AT LOCAL POINT 9
C*    T                 -   POTENTIAL TEMPERATURE AT LOCAL POINT 9
C*    FLDVSQ            -   LOCAL RESIDUE WHEN TEMPERATURE = T
C*    ACSQ              -   LOCAL RESIDUE WHEN VELOCITY COMPONENTS =
C*                          V1,V2
C*    OUTPUTS:
C*    DTT               -   PARTIAL WITH RESPECT TO T
C*    DUT               -   PARTIAL WITH RESPECT TO V1
C*    DVT               -   PARTIAL WITH RESPECT TO V2
C*
C*****************************************************************************
      INCLUDE INPPRM.LIST
      DATA DEL/.05/, DELT/.1/
C
      A0 = ACSQ
      T0 = FLDVSQ
C
C**   VELOCITY DERIVATIVES
C**   CHANGE IN PLUS 1 DIRECTION
      V1T = V1+DEL
      V2T = V2
      CALL ACCSQR(V1T,V2T,A1PT,ADUM)
      A1P = A1PT
C
C**   CHANGE IN PLUS 2 DIRECTION
      V1T = V1
      V2T = V2+DEL
      CALL ACCSQR(V1T,V2T,A2PT,ADUM)
      A2P = A2PT
C
C**   DERIVATIVES
      DUT = (A1P-A0)/DEL
      DVT = (A2P-A0)/DEL
C
      IF(ILEVEL.LE.1) GO TO 70
C
C**   TEMPERATURE DERIVATIVE
C
C**   PLUS CHANGE
      TTP = T+DELT



      CALL TPFLUX(TTP,FTPT)
      TP = FTPT
C
C**   DERIVATIVES
      DTT = (FTPT-T0)/DELT
      RETURN
C
   70 CONTINUE
      DTT = 0.0
      RETURN
      END
```

```
      SUBROUTINE INPUTS
C
C*****************************************************************************
C*
C*    INPUTS TRANSFERS DATA FROM /INTLYZ/ TO /START/, /MRCIPF/ TO
C*    /INPPRM/, AND INITIALIZES IMPORTANT CONSTANTS.
C*
C*    INPUTS:
C*    GENWND(1)
C*    GENWND(2)
C*    DELQ
C*    ILE,ILW,JLN,JLS
C*    IHIERC,IRELAX,IFLUX
C*
C*    OUTPUTS*
C*    EVERYTHING ELSE
C*
C
      INCLUDE MESAGE
      INCLUDE FILES
      INCLUDE MRCIPF
      INCLUDE START
      INCLUDE INPPRM
      INCLUDE INTLYZ
      INCLUDE SETBOX
C
C**   INITITIALIZE:
C
      DGRID = DELTER
      ZGRID = 6.0
      ILEVEL = IHIERC
      NRELAX = IRELAX
      DD = DGRID*SQRT(2.)
      DR = 1./DD
      IL = ILE
      IC = IL-1
      IC1 = ILW
      JL = JLN
      JC = JL-1
      JC1 = JLS
      VE = GENWND(1)
      VN = GENWND(2)
      DELTQ = DELQ
      IWTFLG = IFLUX
      RWTFLG = IWTFLG
      BOXAV = ZGRID/RWTFLG
C
      RETURN
      END
```

```
      SUBROUTINE LOCPNT(K,L,M)
C
C*****************************************************************************
C*
C*    LOCPNT TRANSFERS DATA FROM GLOBAL TO LOCAL ARRAYS AND CALLS
C*    PTPROF TO COMPUTE PROFILE RELATED QUANTITIES.
C*
C*    INPUTS:
C*    M                 -   INDEX OF LOCAL POINT
C*    K,L               -   GLOBAL INDICES OF LOCAL POINT M
C*    SLOPE1(...)       -   FROM /SLOPES/
C*    SLOPE2(...)       -   FROM /SLOPES/
C*    AREAF(...)        -   FROM /SLOPES/
C*    TEMPEX(...)       -   FROM /TEMPEX/
C*    ROUGH(...)        -   FROM /TERDAT/
C*    ACNRML(...)       -   FROM /ACNRML/
C*    OUTPUTS:
C*    HD1(M)            -   IN /LOCAL/      U COMPONENT OF TERRAIN SLOPE
C*    HD2(M)            -   IN /LOCAL/      V COMPONENT OF TERRAIN SLOPE
C*    AR(M)             -   IN /LOCAL/
C*    TH(M)             -   IN /LOCAL/
C*    UL(M)             -   IN /LOCAL/
C*    VL(M)             -   IN /LOCAL/
C*    EN(M)             -   IN /LOCAL/
C*    EN(M)             -   IN /LOCAL/
C*    THD(M)            -   IN /LOCAL/
C*    BUOY(M)           -   IN /LOCAL/
C*    ER(M)             -   IN /LOCAL/
C*    LOCAL VARIABLES:
C*    Z0                -   ROUGHNESS AT POINT M
C*    AN                -   NORMAL ACCELERATION AT POINT M
C*    THEX              -   AMBIENT TEMPERATURE
C*    USQ               -   MAGNITUDE SQUARE OF LOCAL WIND
C*
C*****************************************************************************
C
      INCLUDE GRDPRM
      INCLUDE INPPRM
      INCLUDE FIELDS
      INCLUDE LOCAL
      INCLUDE TERDAT
      INCLUDE SLOPES
      INCLUDE TEMPEX
      INCLUDE ACNRML
C
      HD1(M) = SLOPE1(K,L)
      HD2(M) = SLOPE2(K,L)
      AR(M) = AREAF(K,L)
      Z0 = ROUGH(K,L)
      THEX = TEMPEX(K,L)




      TH(M) = THTA(K,L)
      UL(M) = U(K,L)
      VL(M) = V(K,L)
      AN = ACNRML(K,L)
      USQ = UL(M)**2 +VL(M)**2 +(HD1(M)*UL(M)+HD2(M)*VL(M))**2
      CALL PTPROF(THEX,TH(M),USQ,Z0,AN,  EN(M),THD(M),BUOY(M))
      ER(M) = SQRT(4.*EN(M)+1.)
C
      RETURN
      END
```

```
      SUBROUTINE LOCQUA(I,J)
C
C*****************************************************************************
C*
C*    LOCQUA DOES THE FOLLOWING:
C*    1. DETERMINES THE GLOBAL ARRAY INDICES FOR EACH LOCAL POINT
C*       NEEDED TO COMPUTE LOCAL RESIDUES
C*    2. CALLS LOCPNT TO OBTAIN TERRAIN, TEMPERATURE, AND PROFILE
C*       QUANTITES AT RELEVANT LOCAL POINTS
C*
C*    INPUTS:
C*    I,J               -   GLOBAL INDICES OF POINT WHERE LOCAL RESIDUE
C*                          IS TO BE CALCULATED
C*    NSET              -   FROM /SETBOX/
C*    LOCAL VARIABES:
C*    KINDX(.)          -   ARRAY USED TO DETERMINE GLOBAL INDICES
C*    LINDX(.)          -   ARRAY USED TO DETERMINE GLOBAL INDICES
C*    K,L               -   GLOBAL INDICES OF LOCAL POINTS
C*
C*****************************************************************************
C
      INCLUDE SETBOX
      INCLUDE INPPRM
C
      DIMENSION INDEX(9),KINDEX(9),LINDEX(9)
      DATA KINDEX/1,0,-1,-1,-1,0,1,1,0/
      DATA LINDEX/1,1,1,0,-1,-1,-1,0,0/
C
C**   LOCAL POINT 9 ALWAYS NEEDED
      DATA INDEX(9)/1/
C**   INITILIZE OTHER POINTS AS NOT NEEDED
      DO 1 M=1,8
      INDEX(M) = 0
    1 CONTINUE
C
C**   FIND WHICH POINTS ARE NEEDED
      IF(NSET(1).EQ.0) GO TO 10
      INDEX(1) = 1
      INDEX(2) = 1
      INDEX(8) = 1
C
C
   10 IF(NSET(2).EQ.0) GO TO 20
      INDEX(2) = 1
      INDEX(3) = 1
      INDEX(4) = 1
C
   20 IF(NSET(3).EQ.0) GO TO 30
      INDEX(4) = 1
      INDEX(5) = 1
      INDEX(6) = 1
C
```

```
   30 IF(NSET(4).EQ.0) GO TO 40
      INDEX(6) = 1
      INDEX(7) = 1
      INDEX(8) = 1
C
   40 CONTINUE
C
C**   OBTAIN LOCAL QUANTITIES
      DO 50 M=1,9
      IF(INDEX(M).EQ.0) GO TO 50
C**   SET GLOBAL INDICES CORESPONDING TO LOCAL POINT
      K = I+KINDEX(M)
      L = J+LINDEX(M)
      CALL LOCPNT(K,L,M)
   50 CONTINUE
C
      RETURN
      END
```

```
      SUBROUTINE OUTPUT
C
C*****************************************************************************
C*
C*    OUTPUT BLOCKS OUT THE SAVED WIND AND TEMPERATURE FIELDS TO
C*    MASS STORAGE AND PRINTS A FORMATTED DESCRIPTION OF THE WIND
C*    MODEL RESULTS.
C*
C*    INPUTS:
C*    USAVE(...)          -   FROM /SAVEM/
C*    VSAVE(...)          -   FROM /SAVEM/
C*    TPSAVE(...)         -   FROM /SAVEM/
C*    NSAVE               -   FROM /SAVEM/
C*    NRELAX              -   FROM /INPPRM/
C*    DGRID               -   FROM /INPPRM/
C*    ZGRID               -   FROM /INPPRM/
C*    RES(.)              -   FROM /MNSRCH/
C*    TRES(.)             -   FROM /MNSRCH/
C*    VDRSUM(.)           -   FROM /MNSRCH/
C*    ENERGY(.)           -   FROM /MNSRCH/
C*    RCOUNT                  FROM /MRCHDR/
C*    XWIND(.)                FROM /MRCHDR/
C*    YWIND(.)                FROM /MRCHDR/
C*    DELTAQ(.)               FROM /MRCHDR/
C*    PTEMP(.)                FROM /MRCHDR/
C*    ZELEV(.)                FROM /MRCHDR/
C*    TERBLK(.)               FROM /MRCHDR/
C*    TSITE               -   FROM /INTLYZ/
C*    LOCAL VARIABLES:
C*    IB                  -   BLOCK NUMBER
C*                              = 3*RECOUNT-1 FOR TEMPERATURE
C*                              = 3*RECOUNT FOR U COMPONENT OF WIND
C*                              = 3*RECOUNT+1 FOR V COMPONENT OF WIND
C*
C*
C*****************************************************************************
C
      INCLUDE GRDPRM
      INCLUDE FILES
      INCLUDE MRCIPF
      INCLUDE INPPRM
      INCLUDE MRCHDR
      INCLUDE SAVEM
      INCLUDE MNSRCH
      INCLUDE INTLYZ
      DATA C/.7071067/
C
      WRITE(XOTFIL,10)
      I = RCOUNT
      WRITE(XOTFIL,20) XWIND(I),YWIND(I),DELTAQ(I),
     1     (PTEMP(I,J),ZELEV(I,J),J=1,3),TERBLK(I),TSITE
```

```
      WRITE(XOTFIL,40) DGRID,ZGRID
      WRITE(XOTFIL,50) NRELAX, NSAVE
      WRITE(XOTFIL,60)
      WRITE(XOTFIL,63) (RES(I),I=1,NRELAX)
      WRITE(XOTFIL,61)
      WRITE(XOTFIL,63) (ENERGY(I),I=1,NRELAX)
      WRITE(XOTFIL,62)
      WRITE(XOTFIL,63) (VDRSUM(I),I=1,NRELAX)
C
C
C
C
C**   INVERT SKEW TRANSFORMATION
      DO 120 I = 1,IC
      DO 110 J = 1,JC
      V1 = USAVE(I,J)
      V2 = VSAVE(I,J)
      USAVE(I,J) = C*V1-C*V2
      VSAVE(I,J) = C*V1+C*V2
  110 CONTINUE
  120 CONTINUE
C
C**   ARCHIVE TEMPERATURE AND WIND FIELDS
      IB = 3*RCOUNT - 1
      CALL BLKOUT(NFOUT,TPSAVE(1,1),IB,MOTFIL,IST)
      IB = IB+1
      CALL BLKOUT(NFOUT,USAVE(1,1),IB,MOTFIL,IST)
      IB = IB+1
      CALL BLKOUT(NFOUT,VSAVE(1,1),IB,MOTFIL,IST)
C
C
C**   OUTPUT WIND FIELD
      CALL WNDFLD(USAVE,VSAVE,ILL,JLL)
C
      IF(ILEVEL.LT.2) GO TO 900
C**   PRINT SAVED POTENTIAL TEMPERATURE FIELD
      WRITE(XOTFIL,70)
      WRITE(XOTFIL,63) (TRES(I),I=1,NRELAX)
      CALL TMPFLD(TPSAVE,ILL,JLL)
  900 CONTINUE
C
C**   OUTPUT FORMATS
   10 FORMAT('1   DRIVING PARAMETERS FOR THIS RUN')
   20 FORMAT(//' X WIND =',T40,F8.2,' Y WIND =',T40,F8.2/
     1  1X,'SURFACE HEATING DIFFERENTIAL =       ',T40,F8.2,'  DEG K'//
     2  ' TEMPERATURE PROFILE:',3(T40,F8.2,10X,F8.2)/)/
     3     ' ARCHIVED TERRAIN RUN =',T40,I8/
     4     ' TEMPERATURE AT SITE =',T40,F8.2)
   40 FORMAT('0HORIZONTAL GRID SPACE =',T40,F8.3,' METERS '/,
     1       ' COMPUTATION HEIGHT    =',T40,F8.3,' METERS'//)


   50 FORMAT(1X,I4,' RELAXATION STEPS'/1X,I4,' STEPS TO MINIMUM'//)
   60 FORMAT(//' ACCELERATION SQUARE RESIDUE'/)
   61 FORMAT(//' KINETIC ENERGY'/)
   62 FORMAT(//' DERIVATIVE SQUARE SUM '/)
   63 FORMAT(10(2X,E10.3))
   70 FORMAT(//' TEMPERATURE FLUX RESIDUES'/)
C
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE PTPROF(THEX,THTMP,USQ,Z0,AN,ENT,THDT,BUOY)
C
C**   CALCULATES LOCAL PROFILE RELATED QUANTITIES
C**   WIND PROFILE EXPONENT,ENT-TEMPERATURE DERIVATIVE,THDT-
C**   BUOYANCY ACCELERATION-BUOY
C
      INCLUDE INPPRM
      EQUIVALENCE (Z,EGRID)
C
      DATA G/9.80/
      BUOY = G*(THEX-THTMP)/THEX
      IF(USQ.LE.0.01) GO TO 3
      RICH = BUOY*Z/USQ
      S = -2.*Z*AN/USQ
      RICHC = 2.*S*(1.+S)
      RICH = RICH + RICHC
      GO TO 5
    3 RICH = 100.*BUOY*Z
C**   CHECK RICHARDSON NO. FOR LAMINAR FLOW
    5 IF(RICH.LE.0.25)  GO TO 10
      ENT = 1.
      GO TO 70
C
C**   PROFILE FOR TURBULENT FLOW
   10 IF(RICH.LE.0.1) GO TO 20
      PHIM = 1.75/(1.-4.*RICH)
      GO TO 50
   20 IF(RICH.LE.0.033) GO TO 30
      PHIM = 0.88/(1.-6.*RICH)
      GO TO 50
   30 IF(RICH.LE.-0.01) GO TO 40
      PHIM = 1./(1.-3.*RICH)
      GO TO 50
   40 PHIM = 1./((1.-12.*RICH)**0.25)
C
C**   OBUKHOV SCALED HEIGHT,XI,AND EXPONENT,ENT
   50 IF(RICH.GE.0.0) GO TO 60
      XI = (1.-1./(PHIM**4))/15.
      X = 1./PHIM
      PSI = 2.*ALOG((1.+X)/2.)+ALOG((1.+X*X)/2.)
     1      -2.*ATAN(X+1.5708)
      ENT = PHIM/(ALOG(Z/Z0)-PSI)
      GO TO 70
   60 XI = (PHIM-1.)/4.7
      ENT = PHIM/(ALOG(Z/Z0)+4.7*XI)
C
C**   PROFILE CORRECTION OF TEMPERATURE
   70 DER = (2.*(THEX-THTMP)/Z)*(ENT**2/(2.-ENT**2))
      THDT = Z*DER/((ENT+1.)*(ENT+2.))
C
      TPA = THTMP - .05*Z*DER
      BUOY = G*(THEX-TPA)/THEX
C
      RETURN
      END
```

```
      FUNCTION PRESEX(X0)
C
      COMMON /TPROF/ DUM(3), X(3)
      DIMENSION Y(3)
      DATA Y/85000..70000.,50000./
C
      JJ = 2
      IF(X0.LE.X(JJ)) GO TO 50
      JJ = 3
C
 50   J = JJ-1
      PRESEX = ((Y(JJ)-Y(J))/(X(JJ)-X(J)))*(X0-X(J))+Y(J)
C
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE RELAX
C
C*****************************************************************************
C*
C*    RELAX IS THE WORKHORSE FOR THE WIND AND TEMPERATURE FIELD
C*    ANALYSIS.  RELAX--
C*    1. CALLS ROUTINES TO CALCULATE LOCAL CONTRIBUTIONS TO THE
C*       ACCELERATION AND TEMPERATURE FLUX RESIDUALS
C*    2. ACCUMULATES RESIDUALS
C*    3. CALLS DELTA TO CALCULATE PARTIAL DERIVATIVES OF RESIDUALS
C*       WITH RESPECT TO WINDS AND TEMPERATURES (DELTA IS SKIPPED
C*       FOR PEGGED VALUES OF WIND AND TEMPERATURE)
C*    4. STORES GLOBAL RESIDUES AND TESTS WHETHER TO SAVE FIELDS
C*    5. APPLIES STEEPEST DESCENT CORRECTIONS TO WIND AND TEMPERATURE
C*       FIELDS
C*
C*    INPUTS:
C*    IC,IC1,JC,JC1      -   FROM /INPPRM/
C*    NR,NRELAX          -   FROM /INPPRM/
C*    ILEVEL             -   FROM /INPPRM/
C*    TRLX,VRLX          -   FROM /INPPRM/
C*    RES(.)             -   FROM /MNSRCH/
C*    OUTPUTS:
C*    USAVE(...)         -   IN /SAVEM/
C*    VSAVE(...)         -   IN /SAVEM/
C*    TPSAVE(...)        -   IN /SAVEM/
C*    NSAVE              -   IN /SAVEM/
C*    RES(.)             -   IN /MNSRCH/
C*    TRES(.)            -   IN /MNSRCH/
C*    ENERGY(.)          -   IN /MNSRCH/
C*    VDRSUM(.)          -   IN /MNSRCH/
C*    COMMON LOCATIONS:
C*    U(...)             -   IN /WNDVEL/  FIELDS PROC
C*    V(...)             -   IN /WNDVEL/  FIELDS PROC
C*    THTA(...)          -   IN /POTEMP/  FIELDS PROC
C*    DU(...)            -   IN /DERIV/
C*    DV(...)            -   IN /DERIV/
C*    DT(...)            -   IN /DERIV/
C*    ACNRML             -   IN /ACNRML/
C*    LOCAL VARIABLES:
C*    RSUM               -   ACCELERATION RESIDUAL
C*    RSUMT              -   TEMPERATURE FLUX RESIDUAL
C*    DRSUM              -   SUM OF SQUARES OF ACCELERATION PARTIALS
C*    DRSUMT             -   SUM OF SQUARES OF TEMPERATURE FLUX PARTIALS
C*    SVSQ               -   SUM OF SQUARES OF SURFACE PARALLEL VELOCITY
C*                           COMPONENTS
C*    CFAC               -   RELAXATION FACTOR FOR WIND FIELD
C*    CFACT              -   RELAXATION FACTOR FOR TEMPERATURE FIELD
C*
C*****************************************************************************
C
```

```
      INCLUDE GRDPRM
      INCLUDE INPPRM
      INCLUDE LOCAL
      INCLUDE FIELDS
      INCLUDE SURFOB
      INCLUDE MNSRCH
      INCLUDE DERIV
      INCLUDE SAVEM
      INCLUDE SETBOX
      INCLUDE ACNRML
C
C*****************************************************************
C**   INITIALIZE:
      RSUM = 0.0
      DRSUM = 0.0
      CFAC = 0.0
      RSUMT = 0.0
      DRSUMT = 0.0
      CFACT = 0.0
      SVSQ = 0.0
C
C*****************************************************************
C**   DOUBLE DO LOOP FOR VELOCITY AND TEMPERATURE CORRECTIONS
      DO 60 I=IC1,IC
      DO 50 J=JC1,JC
C
C**   SAVE LOCAL QUANTITIES
      V1 = U(I,J)
      V2 = V(I,J)
      T = THTA(I,J)
C
C**   DETERMINE NEIGHBORING FLUX BOXES
      CALL STGDBX(I,J)
      IF(IUTFLG.EQ.2) CALL WEIGHT(V1,V2)
C
C**   OBTAIN LOCAL QUANTITIES
      CALL LOCQUA(I,J)
C
C**   OBTAIN LOCAL RESIDUES
      CALL ACCSOR(V1,V2,ACSQ,ACNORM)
      ACNRML(I,J) = ACNORM
      IF(ILEVEL.GT.1) CALL TPFLUX(T,FLDVSQ)
      RSUM = RSUM+ACSQ
      RSUMT = RSUMT + FLDVSQ
      IF(NR.EQ.NRELAX) GO TO 50
C
```

```
C**   CHECK IF OBSERVATION POINT
      IF(NUMOBS.EQ.0) GO TO 41
      DO 40 M=1,NUMOBS
C**   IF I,J IS A PEGGED POINT, SKIP DERIVATIVES
      IF(I-KINDX(M)) 40,20,40
   20 IF(J-LINDX(M)) 40,30,40
   30 DU(I,J) = 0.0
      DV(I,J) = 0.0
      DT(I,J) = 0.0
      GO TO 50
   40 CONTINUE
   41 CONTINUE
C
C**   COMPUTE TEMP AND VELOCITY DERIVATIVES
      CALL DELTA(V1,V2,T,FLDVSQ,ACSQ,DTT,DUT,DVT)
      DU(I,J) = DUT
      DV(I,J) = DVT
      DT(I,J) = DTT
      DRSUM = DRSUM + DUT**2 + DVT**2
      DRSUMT = DRSUMT + DTT**2
      SVSQ = SVSQ + U(I,J)**2 + V(I,J)**2
C
   50 CONTINUE
   60 CONTINUE
C
C*************************************************
C
C**   STORE RESIDUES
C**   TEST WHETHER TO SAVE FIELDS
      RES(NR) = RSUM
      TRES(NR) = RSUMT
      ENERGY(NR) = SVSQ
      VDRSUM(NR) = DRSUM
      IF(RES(NR).GT.RES(NSAVE)) GO TO 89
      NSAVE = NR
      DO 85 I=IC1,IC
      DO 85 J=JC1,JC
      USAVE(I,J) = U(I,J)
      VSAVE(I,J) = V(I,J)
   85 TPSAVE(I,J) = THTA(I,J)
   89 CONTINUE
C
      IF(NR.EQ.NRELAX) RETURN
C
C*************************************************
C**   APPLY STEEPEST DESCENT CORRECTIONS
C
```

```
      DRSUM = SQRT(DRSUM)
      DRSUMT = SQRT(DRSUMT)
      IF(DRSUMT.NE.0.0) CFACT = TRLX/DRSUMT
      CFAC = VRLX/DRSUM
      DO 95 I=IC1,IC
      DO 95 J=JC1,JC
      U(I,J) = U(I,J) - DU(I,J)*CFAC
      V(I,J) = V(I,J) - DV(I,J)*CFAC
      THTA(I,J) = THTA(I,J) - DT(I,J)*CFACT
   95 CONTINUE
C-
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE SETDAT
C
C*****************************************************************************
C*
C*    SETDAT COMPUTES TERRAIN SLOPES, AREA FACTORS, AND AMBIENT
C*    TEMPERATURES WHICH ARE REPEATEDLY USED IN THE RELAXATION
C*    PROCESS.
C*
C*    INPUTS:
C*    IC,IC1,JC,JC1      -   FROM /INPPRM/
C*    OUTPUTS:
C*    SLOPE1(...)        -   IN /SLOPES/
C*    SLOPE2(...)        -   IN /SLOPES/
C*    AREAF(...)         -   IN /SLOPES/
C*    TEMPEX(...)        -   IN /TEMPEX/
C*
C*****************************************************************************
C
      INCLUDE INPPRM
      INCLUDE GRDPRM
      INCLUDE TERDAT
      INCLUDE SLOPES
      INCLUDE TEMPEX
C
      DO 100 I=IC1,IC
      DO 100 J=JC1,JC
      SLOPE1(I,J) = (H(I+1,J+1)-H(I,J))*DR
      SLOPE2(I,J) = (H(I,J+1)-H(I+1,J))*DR
      AREAF(I,J) = SQRT(1.+SLOPE1(I,J)**2+SLOPE2(I,J)**2)
      TEMPEX(I,J) = TEXF(H(I,J))
  100 CONTINUE
C
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE SETUP
C
C*****************************************************************************
C*
C*    SETUP INITIALIZES THE MICRO SCALE WIND AND TEMPERATURE FIELDS
C*    USING MESOSCALE INPUTS AND SPOT OBSERVATIONS.
C*    PARAMETERS ARE ALSO INITIALIZED.
C*
C*    INPUTS:
C*    VE                -   MESOSCALE
C*    VN                -     DRIVERS FROM
C*    DELTQ             -       /START/
C*    ILEVEL            -   FROM /INPPRM/
C*    IC,IC1,JC,JC1     -   FROM /INPPRM/
C*    KINDX(.)          -   FROM /SURFOB/
C*    LINDX(.)          -   FROM /SURFOB/
C*    UW(.)             -   FROM /SURFOB/
C*    VW(.)             -   FROM /SURFOB/
C*    THETAO(.)         -   FROM /SURFOB/
C*    NUMOBS            -   FROM /SURFOB/
C*    OUTPUTS:
C*    U(...)            -   IN /WNDFLD/  FIELDS PROC
C*    V(...)            -   IN /WNDFLD/  FIELDS PROC
C*    THETAO(...)       -   IN /POTEMP/
C*    VMAGSQ            -   IN /DRIVER/
C*    VMAG              -   IN /DRIVER/
C*    VRLX              -   IN /INPPRM/
C*    TRLX              -   IN /INPPRM/
C*
C*****************************************************************************
C
      INCLUDE GRDPRM
      INCLUDE INPPRM
      INCLUDE TEMPEX
      INCLUDE START
      INCLUDE DRIVER
      INCLUDE SURFOB
      INCLUDE FIELDS
      INCLUDE TERDAT
      INCLUDE ROTATE
C
      DATA P2/.2862/
C
      ANG = ROTATE + .7853981
      C1 = COS(ANG)
      C2 = SIN(ANG)
C
C
C**   INITIALIZE WIND AND TEMPERATURE FIELDS AND INPUT PARAMETERS
      VMAGSQ = VE**2+VN**2
      VMAG = SQRT(VMAGSQ)
      CALL SETDAT
```

```
      UB = C1*VE+C2*VN
      VB = -C2*VE+C1*VN
      DO 20 I = IC1,IC
      DO 10 J = JC1,JC
      U(I,J) = UB
      V(I,J) = VB
      THTA(I,J) = TEMPEX(I,J)+DELTO*(100000.0/PRESEX(H(I,J)))**P2
   10 CONTINUE
   20 CONTINUE
C
C**   SET RELAXATION SCALING PARAMETERS
      PTFAC = SQRT((IC+1-IC1)*(JC+1-JC1))
      VRLX = .02*PTFAC*AMAX0(VMAG,2.)
      TRLX = .1*PTFAC
C
C**   SET PEGGED OBSERVATIONS
      IF(NUMOBS.EQ.0) GO TO 150
      DO 100 M=1,NUMOBS
      K = KINDX(M)
      L = LINDX(M)
      U(K,L) = UU(M)
      V(K,L) = VV(M)
      THTA(K,L) = THETA0(M)
  100 CONTINUE
  150 CONTINUE
      IF(ILEVEL.LT.2) RETURN
      CALL TMFFLD(THTA,ICC,JCC)
C
      RETURN
      END
```

```
      SUBROUTINE STGDBX(I,J)
C
C*****************************************************************************
C*
C*    STGDBX DETERMINES WHETHER AN ARRAY POINT IS IN THE INTERIOR
C*    OR ON THE BOUNDARY OF THE ANALYSIS REGION, AND SETS THE FLAGS
C*    NSET(.) TO IDENTIFY THE LOCAL FLUX BOXES WHICH MAY BE ASSOCIATED
C*    WITH THE ARRAY POINT.
C*
C*    INPUTS:
C*    I,J               -   LOCATION INDICES
C*    IC,IC1,JC,JC1     -   FROM /INPPRM/
C*    OUTPUTS:
C*    NSET(.)           -   IN /SETBOX/
C*
C*****************************************************************************
C
      INCLUDE INPPRM
      INCLUDE SETBOX
C
      DO 4 M = 1,4
      NSET(M) = 1
    4 CONTINUE
      IF(I.NE.IC1) GO TO 6
      NSET(2) = 0
      NSET(3) = 0
    6 IF(I.NE.IC) GO TO 8
      NSET(1) = 0
      NSET(4) = 0
    8 IF(J.NE.JC1) GO TO 10
      NSET(3) = 0
      NSET(4) = 0
   10 IF(J.NE.JC) GO TO 12
      NSET(1) = 0
      NSET(2) = 0
   12 CONTINUE
      RETURN
      END
```

```
      FUNCTION TEXF(X0)
C
C**   THIS ROUTINE EXTRAPOLATES A VALUE OF SURFACE AMBIENT
C**   POTENTIAL TEMPERATURE FROM THE UPPER AIR PROFILE
C
      PARAMETER LEVELS=3
      COMMON /TPROF/ Y(LEVELS), X(LEVELS)
C
      L1 = LEVELS-1
      DO 30 I=2,L1
      JJ = I
      IF(X0.LE.X(JJ)) GO TO 50
 30   CONTINUE
      JJ = LEVELS
C
 50   J = JJ-1
      TEXF = ((Y(JJ)-Y(J))/(X(JJ)-X(J)))*(X0-X(J))+Y(J)
C
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE TMPFLD(TPSAVE,ILL,JLL)
C
      INCLUDE INPPRM
      INCLUDE FILES
      DIMENSION TPSAVE(ILL,JLL)
C
C**   OUTPUT POTENTIAL TEMPERATURE FIELD
C
      WRITE(XOTFIL,80)
      ILIM = (IC+1-IC1)/12 + 1
      DO 800 II=1,ILIM
      I1 = 12*(II-1)+IC1
      I2 = 12*II + IC1-1
      IF(I2.GT.IC) I2 = IC
      IF(I1.GT.IC) GO TO 800
      IF(II.NE.1) WRITE(XOTFIL,71)
      DO 700 JJ=JC1,JC
      J = JC+1-JJ
      WRITE(XOTFIL,82) (TPSAVE(I,J),I=I1,I2)
  700 CONTINUE
  800 CONTINUE
C
   71 FORMAT(///' NEXT SET OF COLUMNS'//)
   80 FORMAT(1H1,3X,'POTENTIAL TEMPERATURE FIELD'//)
   82 FORMAT(12(3X,F7.1))
C
      RETURN
      END
```

```
      SUBROUTINE TPFLUX(T,FLDVSQ)
C
C*****************************************************************************
C*
C*    TPFLUX OBTAINS ESTIMATES OF TEMPERATURE FLUX RESIDUALS FROM LOCAL
C*    BOXES, AVERAGES THEM, APPLIES PROPER SCALE FACTORS, AND RETURNS A
C*    VALUE TO RELAX.
C*    INPUTS:
C*    NSET(.)            -   FROM /SETBOX/
C*    T                  -   TEMPERATURE AT LOCAL POINT 9
C*    BOXAV              -   FROM /SETBOX/
C*    OUTPUTS:
C*    FLDVSQ             -   TEMPERATURE FLUX RESIDUAL
C*
C*****************************************************************************
C
      INCLUDE LOCAL
      INCLUDE INPPRM
      INCLUDE SETBOX
C
      TH(9) = T
      FLDVSQ = 0.0
C
      DO 100 I=1,4
      IF(NSET(I).EQ.0) GO TO 100
      CALL TPFLXL(I,TFLUXL)
      FLDVSQ = FLDVSQ + TFLUXL
  100 CONTINUE
C
      FLDVSQ = FLDVSQ*BOXAV/AR(9)
      RETURN
      END
```

```
      SUBROUTINE TPFLXL(IPT,TFLUXL)
C
C*****************************************************************************
C*
C*    TPFLXL ESTIMATES THE TEMPERATURE FLUX RESIDUAL (UP TO A FACTOR)
C*    CONTRIBUTED BY A SINGLE LOCAL BOX.
C*
C*    INPUTS:
C*    IPT               -   INDEX OF LOCAL BOX (= 1, 2, 3, OR 4)
C*    UL(.),VL(.)       -   FROM /LOCAL/
C*    TH(.),THD(.)      -   FROM /LOCAL/
C*    OUTPUTS:
C*    TFLUXL            -   ESTIMATE OF RESIDUAL
C*    LOCAL VARIABLES:
C*    IPOINT(...)       -   SETS WHICH OF 9 LOCAL POINTS CORRESPOND TO
C*                          LOCAL BOX IPT
C*    I1,I2,I3,I4       -   INDICES OF LOCAL POINTS
C*    TAV               -   AVERAGE TEMPERATURE
C*    TF                -   TEMPERATURE FLUX
C*
C*****************************************************************************
C
      INCLUDE LOCAL
      INCLUDE INPPRM
      DIMENSION IPOINT(4,4)
      DATA IPOINT /1,2,9,8,2,3,4,9,9,4,5,6,8,9,6,7/
C
      I1 = IPOINT(1,IPT)
      I2 = IPOINT(2,IPT)
      I3 = IPOINT(3,IPT)
      I4 = IPOINT(4,IPT)
C
      TAV = (TH(I3)+TH(I1)+TH(I2)+TH(I4))/4.0
      TF = (UL(I1)*(TH(I1)-TAV)/(EN(I1)+1.)-UL(I1)*THD(I1))*AR(I1)
     1    +(VL(I2)*(TH(I2)-TAV)/(EN(I2)+1.)-VL(I2)*THD(I2))*AR(I2)
     2    -(UL(I3)*(TH(I3)-TAV)/(EN(I3)+1.)-UL(I3)*THD(I3))*AR(I3)
     3    -(VL(I4)*(TH(I4)-TAV)/(EN(I4)+1.)-VL(I4)*THD(I4))*AR(I4)
C
      TFLUXL = TF**2
C
      RETURN
      END
```

```
      SUBROUTINE WEIGHT(V1,V2)
C
C*****************************************************************************
C*
C*    WEIGHT ALTERS THE FLAGS NSET(.) SO THAT UPSTREAM BOXES ARE USED
C*    TO ESTIMATE THE LOCAL CONTRIBUTIONS TO THE ACCERATION AND
C*    TEMPERATURES RESIDUES
C*
C*    INPUTS:
C*    V1,V2              -   COMPONENTS OF LOCAL WIND VECTOR
C*    OUTPUTS:
C*    NSET(.)            -   IN /SETBOX/
C*
C*****************************************************************************
C
      INCLUDE SETBOX
C
      DO 1 I=1,4
      IF(NSET(I).EQ.0) RETURN
    1 CONTINUE
C
C**   DETERMINE UPSTREAM BOXES
      IF(V1) 6,7,7
    6 NSET(3) = 0
      GO TO 8
    7 NSET(1) = 0
C
    8 IF(V2) 9,10,10
    9 NSET(4) = 0
      RETURN
   10 NSET(2) = 0
      RETURN
C
      END
```

```
      SUBROUTINE WINDEX
C
C*****************************************************************************
C*
C*    WINDEX IS THE MANAGER FOR MRC MICRO WIND MODEL
C*
C*    INPUTS:
C*    NRELAX            -   FROM  /INPPRM/
C*    NR                -   FROM  /INPPRM/
C*
C*****************************************************************************
C
C
      INCLUDE GRDPRM
      INCLUDE MESAGE
      INCLUDE FILES
      INCLUDE MRCIPF
      INCLUDE SAVEM
      INCLUDE MNSRCH
      INCLUDE TEMPEX
      INCLUDE SLOPES
      INCLUDE START
      INCLUDE INPPRM
      INCLUDE INTLYZ
      INCLUDE TERDAT
      INCLUDE SETBOX
      INCLUDE DRIVER
C
C
C
C**   INIITIALIZE:
      CALL INPUTS
C
C
C**   INITIALIZE WIND AND TEMP FIELDS. SKEW TRANSFORM
      CALL SETUP
C
C
C**   ADJUST WIND AND TEMPERATURE FIELDS
      NSAVE = 1
      DO 50 I=1,NRELAX
      NR = I
      CALL RELAX
   50 CONTINUE
C
C
C**   OUTPUT RESULTS
      CALL OUTPUT
C
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE WNDFLD(USAVE,VSAVE,ILL,JLL)
C
      INCLUDE INPPRM
      INCLUDE FILES
      DIMENSION USAVE(ILL,JLL), VSAVE(ILL,JLL)
C
C
C**   OUTPUT WIND FIELD
C
      WRITE(XOTFIL,70)
      ILIM = (IC+1-IC1)/8 + 1
      DO 600 II=1,ILIM
      I1 = 8*(II-1)+IC1
      I2 = 8*II + IC1-1
      IF(I2.GT.IC) I2 = IC
      IF(I1.GT.IC) GO TO 600
      IF(II.NE.1) WRITE(XOTFIL,71)
      DO 500 JJ=JC1,JC
      J = JC+JC1-JJ
      WRITE(XOTFIL,72) (USAVE(I,J),VSAVE(I,J),I=I1,I2)
  500 CONTINUE
  600 CONTINUE
C
   70 FORMAT(1H1,3X,' WIND FIELD'//)
   71 FORMAT(///' NEXT SET OF COLUMNS'//)
   72 FORMAT(8(3X,F5.1,',',F5.1))
C
      RETURN
      END
```

D.4 JOB CONTROL ELEMENTS AND SAMPLE INPUT OF MRC*EPAMS

```
***   JCL ELEMENTS USED FOR FILE AND LOGICAL UNIT ASSIGNMENTS
*** M.WNDASG/SWUS
@USE F3.,MRC*TERRAIN.
@ASG,AZ MRC*TERRAIN.
@USE 10.. XINFIL.
@ASG,AZ  XINFIL.
@USE F8..SWOBS.
@ASG,AZ SWOBS.
@USE F1.,SWUAD.
@ASG,AZ SWUAD.
@USE F12..SWGWCD.
@ASG,AZ SWGWCD.
@USE F11..SWGWCP.
@ASG,AZ SWGWCP.
@USE F15..SWUS*SFDTA1.
@ASG,AZ SWUS*SFDTA1
@USE F16..SWUS*MRC TRF.
@ASG,AZ F16.
@USE F17.,SWUSMRC*WNDARCHIVE.
@ASG,AZ SWUSMRC*WNDARCHIVE.
@USE F7..323*WSGPD2.
@ASG,AZ F7.
***   M.WNDASG/EUR
@USE 10.. XINFIL.
@ASG,AZ  XINFIL.
@USE F8..EUROBS.
@ASG,AZ EUROBS.
@USE F1..EURUAD.
@ASG,AZ EURUAD.
@USE F15..EURO*SFDTA1.
@ASG,AZ F15.
@USE F16..EURO*MRCTRF.
@ASG,AZ F16.
@USE F17.,EUROMRC*WNDARCHIVE.
@ASG,AZ F17.
```

```
***   INPUT ELEMENT FOR MAP PROCESSOR --- M.WNDMAP/SWUS
@MAP,A   .M.BIGKAHUNA/SWUS
  SEG XESEG
  IN M.XECAMS.ASL*UTILITY.BLOCKS
    SEG A*.(XESEG)
    IN M.ANALYZ
    SEG B*.(XESEG)
    IN M.PLTPRG
    SEG C*.(XESEG)
    IN M.APPLY
      SEG H*.(A)
      IN M.HEC..MIXLYR
      SEG W*.(A)
      IN M.WNDMGR.M.IDIFTM
        SEG DT*.(W)
        IN M.DTBMGR/SWUS..CLSSTN/SWUS,T.SATVP,M.INTPP
        SEG WI*.(W)
        IN M.WINDEX/SWUS..TEXF..OUTPUT/SWUS..PRESEX..INPUTS
        IN M.WNDFLD.M.TMPFLD
        SEG TP*.(W)
        IN M.TERACQ/SWUS
        SEG DR*.(W)
        IN M.DRIVRS/SWUS
          SEG UA*.(DT)
          IN M.MRCUA/SWUS..UADATA/SWUS,T.XTRCUA
          SEG SFC*.(DT)
          IN M.MRCSFC/SWUS,M.SFCDTA/SWUS.ASL*UTILITY.TESTFD
          SEG GW*.(DT)
          IN M.MRCGWC/SWUS,M.GWCDTA/SWUS
          SEG MD*.(DT)
          IN M.MESMOD/SWUS
          SEG NET*.(DT)
          IN M.MRCNET/SWUS
          SEG DA*.(DT)
          IN M.DATANL/SWUS..MINV..INTERP..INTCOF..TMPPRF
            SEG 01*.(DA)
            IN M.MANUAL/SWUS
            SEG 02*.(DA)
            IN M.NETVAR
          SEG 03*.(DA)
            IN M.MESVAR
            SEG 04*.(DA)
            IN M.PRUFLS/SWUS
            SEG 05*.(DA)
            IN M.SRFTMP/SWUS
            SEG 06*.(DA)
            IN M.UAVAR/SWUS..GEOWND
          SEG SETSEG*.(WI)
          IN M.SETUP/SWUS. .SETDAT/SWUS
          SEG PX*.(WI)
          IN M.RELAX/SWUS..STGDBX..LOCOUA..LOCPNT/SWUS..PTPROF
          IN M.ACCSOR..ACCSOL..DELTA..WEIGHT
    IN M.TPFLUX..TPFLXL
  END
```

```
***   SAMPLE NAMELIST INPUT ELEMENTS
***   M.WNDINP/SWUS
 $XFLAGS  FUNCTN=2, TASK=2, JOB=0, BLOCK=1, FILE=6, PRINT=-1, LEVEL=1
 $END
 $FLAGS   IDAT(1)=0, IDAT(2)=0, IDAT(3)=0,
     IDAT(4)=1, IDAT(5)=1, IDAT(6)=1, IMDT(1)=0, IMDT(2)=2, IMDT(3)=0,
     IMDT(4)=8, IMDT(5)=0, IMDT(6)=0,  NTST=0, IFLUX=2, ILASTR=1,
     IHIERC=2, MTRBLK=9, IRELAX=50, ILE=40, JLN=40, ILW=1,JLS=1
 $END
 $TIMPLA  XPLACE=351.3, YPLACE=3585.7, ZPLACE=2487.0, YEAR=1974,
     MONTH=11, DAY= 1, HOUR=14, MINUTE=00
 $END
 $XFLAGS  FUNCTN=2, TASK=2, JOB=0, BLOCK=1, FILE=6, PRINT=-1, LEVEL=1
 $END
 $FLAGS   IDAT(1)=0, IDAT(2)=0, IDAT(3)=0,
     IDAT(4)=1, IDAT(5)=1, IDAT(6)=1, IMDT(1)=0, IMDT(2)=2, IMDT(3)=0,
     IMDT(4)=8, IMDT(5)=0, IMDT(6)=0,  NTST=0, IFLUX=2, ILASTR=1,
     IHIERC=2, MTRBLK=9, IRELAX=50, ILE=40, JLN=40, ILW=1,JLS=1
 $END
 $TIMPLA .XPLACE=351.3, YPLACE=3585.7, ZPLACE=2487.0, YEAR=1974,
     MONTH=11, DAY= 2, HOUR=14, MINUTE=00
 $END
 $TIMPLA  XPLACE=351.3, YPLACE=3585.7, ZPLACE=2487.0, YEAR=1974,
     MONTH=11, DAY= 5, HOUR=14, MINUTE=00
 $END
***   M.WNDINP/EUR
 $XFLAGS  FUNCTN=2, TASK=2, JOB=0, BLOCK=1, FILE=6, PRINT=-1, LEVEL=1
 $END
 $FLAGS   IDAT(1)=0, IDAT(2)=0, IDAT(3)=1,
     IDAT(4)=1, IDAT(5)=1, IDAT(6)=1, IMDT(1)=1, IMDT(2)=2, IMDT(3)=0,
     IMDT(4)=8, IMDT(5)=0, IMDT(6)=0,  NTST=0, IFLUX=2, ILASTR=1,
     IHIERC=2, MTRBLK=1, IRELAX=50, ILE=51, JLN=31, ILW=1,JLS=1
 $END
 $TIMPLA  XPLACE=351.3, YPLACE=3585.7, ZPLACE=0.0, YEAR=1974,
     MONTH=11, DAY= 2, HOUR= 7, MINUTE=00
 $END
```

D.5 PLOT PROGRAMS OF MRC*UTILITY.

D.5.1 Procedure Elements of Plot Programs (PLTPRG)

```
FILES   PROC
C
      COMMON /FILES/ XINFIL,XOTFIL,MICTRF,MOTFIL
      INTEGER XINFIL,XOTFIL
C
 END
MESAGE  PROC
      COMMON /MESAGE/ FUNCTN, TASK, JOB, BLOCK, FILE, PRINT, LEVEL
      INTEGER FUNCTN, TASK, JOB, BLOCK, FILE, PRINT
C
 END
MRCHDR  PROC
      PARAMETER RUNS=25, NPARAM=6
      PARAMETER HDRSIZ=(10+NPARAM)*RUNS+1
      COMMON /MRCHDR/ RCOUNT,PTEMP(RUNS,3),ZELEV(RUNS,3),
     1                DELTAQ(RUNS),XWIND(RUNS),YWIND(RUNS),TERBLK(RUNS)
     2                ,PARAMS(RUNS,NPARAM)
      INTEGER RCOUNT
      INTEGER TERBLK
      INTEGER PARAMS
 END
HOWPLT  PROC
      COMMON /HOWPLT/ WATFIL,ISTASH,ILL,JLL,NGRD,ILE,ILW,JLN,JLS,IPR,
     1 JPR,ITYPE,NUMUP,NUMPLT,PLH,WATPEN,OVRLAY,NCON,OLDTOP,
     2                SAMPLE,TEXT
C
      INTEGER TEXT(6)
      INTEGER WATFIL,WATPEN,OVRLAY,SAMPLE
C
C     /HOWPLT/ CONTAINS FLAGS READ (OR DEFAULTED) BY ROUTINE OMEXEC
C     WHICH SPECIFY THE TYPE OF PLOTTING JOB TO BE DONE
C
C     WATFIL          - LOGICAL UNIT NUMBER OF FILE CONTAINING DATA
C                       TO BE PLOTTED
C     NUMUP           - NUMBER OF PLOTS TO BE VERTICALLY STACKED
C                       (SHOULD BE 1,2, OR 3: MUST BE CONSISTENT
C                       WITH PLH)
C     PLH             - SIZE (INCHES) OF PLOT AXES  (MAX = 27.0)
C     NUMPLT          - COUNTER FOR NUMBER OF PLOTS EXECUTED
C     IPR             -   DIMENSIONS OF DATA ARRAYS:
C     JPR             -   EG UWIND(IPR,JPR)
C     WATPLT          - NUMBER OF BLOCK IN WATFIL CONTAINING DATA
C                         TO BE PLOTTED
C     WATPEN          - PEN NUMBER:  1 - BLACK
C                                    2 - RED
C                                    3 - BLUE
C     OLDORI          - FLAG WHICH CONTROLS OVERLAY PLOTS
C                       OLDORI=1:  PLOT IS OVERLAID ON MOST RECENT PLOT
C                       OLDORI=0:  PLOT IS NOT OVERLAID
C     NCON            - NUMBER OF CONTOUR LINES TO BE DRAWN
C                       IF NCON.LT.0, DEFAULT CONTOUR VALUES ARE COMPUTED
C     ITYPE           - TYPE OF PLOT
C                         = 1  CONTOUR PLOT OF TERRAIN
C                         = 2  CONTOUR PLOT OF TEMPERATURE
C                         = 3  STREAMLINES OF WINDFIELD WITH SPEED
C                              CONTOURS
C                         = 4  VECTOR PLOT OF WINDFIELD WITH SCALED
C                              MAGNITUDES
C                         = 5  THREE DIMENSIONAL PLOT OF TERRAIN
C     TEXT            - ARRAY OF ALPHANUMERIC DATA TO BE USED AS PLOT
C                       TITLE (30 FIELD DATA CHARACTERS)
C     SCALE           - SCALING FACTOR APPLIED TO DATA BEFORE PLOTTING
C                       (IMPROVES APPEARANCE OF CONTOUR VALUES)
```

```
WORK1   PROC
      DIMENSION  WORK1(2601)
 END
CONV    PROC
      COMMON /CONVAL/ CONV(20)
 END
WORK2   PROC
      COMMON/WORK2/ WORK2(2601)
 END
WORK3   PROC
      COMMON/WORK3/ WORK3(2601), WORK4(2601), WORK5(2601)
 END
TBLHDR  PROC
      PARAMETER IDIM=50, THDRSZ=251
      COMMON /TBLHDR/ NNBLK,NDIMXH(IDIM),NDIMYH(IDIM),DELH(IDIM),
     1                SWUTMX(IDIM),SWUTMY(IDIM)
 END
```

D.5.2 Subroutines of Plot Program (PLTPRG)

```
      PROGRAM XECAMT
      INCLUDE MESAGE
      INCLUDE FILES
C
C
      DATA XINFIL/10/, XOTFIL/6/, MOTFIL/17/, MICTRF/16/
C
      CALL PLTPRG
      END



      SUBROUTINE PLTPRG
      CALL OMEXEC
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE OMEXEC
C
      INCLUDE MESAGE
      INCLUDE FILES
      INCLUDE MRCHDR
      INCLUDE HOWPLT
      INCLUDE TBLHDR
      DATA IBLANK/'      '/, IDTITL/'TITLE '/
C
      NAMELIST /OMPLOT/ WATFIL,ISTASH,ILL,JLL,ILE,ILW,JLN,JLS,ITYPE,PLH,
     1                  NUMUP,WATPEN,SAMPLE,OVRLAY,NCON,TEXT,ISTOP
C
C**   BLOCK IN TERRAIN AND WIND ARCHIVE HEADERS:
      CALL BLKIN(THDRSZ,NNBLK,1,MICTRF,IST)
      CALL BLKIN(HDRSIZ,RCOUNT,1,MOTFIL,IST)
C
C**   INITIALIZE PLOT PARAMETERS AND PLOTTER:
      NUMPLT = 0
      ISTAK = 0
      CALL GOPLOT
      WRITE(XOTFIL,1000)
C
C**   SUPPLY DEFAULT VALUES:
      WATFIL = MOTFIL
      ILL    = 40
      JLL    = 40
      ILE    = 40
      ILW    =  1
      JLN    = 40
      JLS    =  1
      PLH    = 25.0
      NUMUP  =  1
      WATPEN =  3
      NCON   =  8
      SAMPLE = 1
      OVRLAY =  0
      ISTOP  =  0
C
   25 CONTINUE
      DO 30 I=2,6
      TEXT(I) = IBLANK
   30 CONTINUE
      TEXT(1) = IDTITL
C
C**   READ INPUT
      READ(XINFIL,OMPLOT,END=150)
      IF(ISTOP.EQ.1) GO TO 150
      NGRD = ILL*JLL
      WRITE(XOTFIL,2000) WATFIL,ISTASH,ILL,JLL,ITYPE,WATPEN,NUMUP,PLH,
```

```
     1                    OVRLAY,NCON,SAMPLE
C
      CALL RESET(ISTAK)
C**   BRANCH TO INDICATED PLOT JOB
      IF(ITYPE.EQ.1) CALL TERCN
      IF(ITYPE.EQ.2) CALL TMPCN
      IF(ITYPE.EQ.3) CALL STRMCN
      IF(ITYPE.EQ.4) CALL PLOVEC
      IF(ITYPE.EQ.5) CALL THREED(ISTAK)
C
      WRITE(XOTFIL,3000) ILE,ILW,JLN,JLS,IPR,JPR
      WRITE(XOTFIL,4000) TEXT,NUMPLT
C
      GO TO 25
  150 CONTINUE
C
      CALL ENDPLT
      RETURN
C
 1000 FORMAT('1START PLOT JOB:  OMEXEC'//)
 2000 FORMAT(' NAMELIST INPUT:'/' FILE NUMBER:',T25,'WATFIL =',T40,I8/
     1       ' ARCHIVED RUN NUMBER:',T25,'ISTASH =',T40,I8/
     2       ' BLOCK IN DIMENSIONS:',T25,'ILL =',T40,I8/T25,'JLL =',T40,
     3         I8/' TYPE OF PLOT:',T25,'ITYPE =',T40,I8/
     4            ' PEN COLOR:',  T25,'WATPEN =',T40,I8/
     5            ' STACKING FLAG:',T25,'NUMUP =',T40,I8/
     6            ' PLOT HEIGHT:',T25,'PLH =',T40,F8.1/
     7            ' OVERLAY FLAG:',T25,'OVRLAY =',T40,I8/
     8            ' NUMBER OF CONTOURS:', T25,'NCON =',T40,I8/
     9            ' SAMPLING FLAG:',      T25,'SAMPLE =',T40,I8//)
 3000 FORMAT(/' GRID DIMENSIONS:',T25,'ILE =', T40,I8/
     1                            T25,'ILW =', T40,I8/
     2                            T25,'JLN =', T40,I8/
     3                            T25,'JLS =', T40,I8/
     4        ' ACTUAL PLOT DIMENSIONS:',T25,'IPR =',T40,I8/
     5                                   T25,'JPR =',T40,I8//)
C
 4000 FORMAT(/' PLOT TITLE:' ,T25,6A6/' PLOT NUMBER ',I2,' COMPLETED'//)
C
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE RESET(ISTAK)
C
      INCLUDE FILES
      INCLUDE HOWPLT
C
C**   THIS ROUTINE RESETS THE ORIGIN FOR EACH PLOT JOB
C**   CONTROLLING PARAMETERS ARE THE OVERLAY FLAG,STACKING FLAG,
C**   NUMBER OF PLOTS, AND PLOT HEIGHT.
C
      IF(OVPLAY.EQ.1) RETURN
      II = ISTAK/NUMUP
      ISTAK = ISTAK-II*NUMUP
      IF(ISTAK.EQ.0) CALL RSTR(2)
      IF(ISTAK.NE.0) CALL PLOT(0.,OLDTOP+1.0,-3)
      NUMPLT = NUMPLT + 1
      ISTAK = ISTAK + 1
C
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE TERCN
C
      INCLUDE FILES
      INCLUDE HOWPLT
      INCLUDE TBLHDR
      INCLUDE WORK1
      INCLUDE CONV
      INCLUDE WORK2
C
C**   DETERMINE FILE BLOCK NUMBER FROM FILE RUN NUMBER
      IBLOCK = 2*ISTASH
C
C**   BLOCK IN AND PREP DATA
      CALL DPREP1
      CALL DPREP2(WORK2,IBLOCK)
C
C**   INITIALIZE PLOTTER AND SET PLOT PARAMETERS
      CALL PLOT(1.0,1.0,3)
      NDEC = -1
      PLTHT = PLH
      NCONT = -NCON
      NSM   = 3
      CALL NEWPN(WATPEN)
C
C**   CALL CONTOUR PLOTTING ROUTINE
      CALL KSOCON(WORK2,WORK1,IPR,JPR,PLTHT,NCONT,CONV,NDEC,NSM)
C
C**   SET OLDTOP FOR POSSIBLE STACKING
      OLDTOP = PLH + 3.5
C
C**   ENCODE AND PLOT TITLE
      ENCODE(36,2000,TEXT(1),ID,ERR=200) NUMPLT,ISTASH
      XT = 1.0
      YT = PLH + 2.0
      CALL NEWPN(2)
      IF(OVRLAY.NE.1) CALL SYMBOL(XT,YT,.15,TEXT(1),0.0,36)
      RETURN
C
  200 WRITE(XOTFIL,1000)
      RETURN
C
 1000 FORMAT(/' TITLE ENCODING ERROR'//)
 2000 FORMAT('TERRAIN:  PLOT # = ',I2,'  TER # = ',I2)
C
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE PLOVEC
C
      INCLUDE HOWPLT
      INCLUDE FILES
      INCLUDE MRCHDR
      INCLUDE WORK2
      INCLUDE WORK3
C
      DIMENSION X(1),Y(1),VX(1),VY(1)
      EQUIVALENCE (WORK2(1),X(1)), (WORK3(1),Y(1)), (WORK4(1),VX(1)),
     1            (WORK5(1),VY(1))
C
      DATA SX,SY,SUV,ANGL /1.0,1.0,1.0,15./
C
C**   COMPUTE BLOCK NUMBER FROM RUN NUMBER
      IBLOCK = 3*ISTASH
C
      CALL DPREP1
      IPR = IPR-1
      JPR = JPR-1
      XINT = PLH/FLOAT(JPR)
C
      CALL DPREP3(IPR,JPR,X,Y,VX,VY,IBLOCK,XINT)
C
C
C**   INITIALIZE PLOTTER AND SET PLOT PARAMETERS
      CALL NEWPN(WATPEN)
      CALL PLOT(1.0,1.0,-3)
      CALL PLOT(X(1),Y(1),3)
      NVEC = IPR*JPR
      ALEN = .1*XINT
C
      CALL VECPLT(X,Y,VX,VY,NVEC,SX,SY,SUV,ALEN,ANGL)
C
      CALL PLOT(-1.0,-1.0,-3)
C
C**   SET OLDTOP FOR POSSIBLE STACKING
      OLDTOP = PLH + 3.5
C**   ENCODE AND PLOT TITLE
      ENCODE(36,3000,TEXT(1),ID,ERR=200) DELTAQ(ISTASH),
     1           TERBLK(ISTASH), ISTASH
      XT = 1.0
      YT = PLH + 2.0
      CALL NEWPN(2)
      IF(OVRLAY.NE.1) CALL SYMBOL(XT,YT,.15,TEXT(1),0.0,36)
      ENCODE(36,2000,TEXT(1),ID,ERR=200) XWIND(ISTASH),YWIND(ISTASH)
      XT = 1.0
      YT = PLH + 2.25
      IF(OVRLAY.NE.1) CALL SYMBOL(XT,YT,.15,TEXT(1),0.0,36)
C
      RETURN
  200 WRITE(XOTFIL,1000)
      RETURN
C
 1000 FORMAT(/' TITLE ENCODING ERROR'//)
 2000 FORMAT('DRIVING WINDS:  VE= ',F5.1,'  VN= ',F5.1)
 3000 FORMAT('BUOY =',F6.1,'  TER # = ',I2,'  WND # = ',I2)
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE THREED(ISTAK)
C
      INCLUDE TBLHDR
      INCLUDE FILES
      INCLUDE HOWPLT
      INCLUDE WORK2
      INCLUDE WORK3
C
      DATA  ANGV/40.0/, FLAT/-1.0/, EYE/4.0/
      REAL ANGH(8)/0.0,45.0,90.0,135.0,180.0,225.0,270.0,315.0/
C
      ISTAK = 0
      CALL DPREP1
      IQ = NCON
      IF((IQ.LT.1).OR.(IQ.GT.8)) IQ = 6
C
      IBLOCK = 2*ISTASH
      CALL DPREP2(WORK2,IBLOCK)
C**   FIND MINIMUM ELEVATION
      N = IPR*JPR
      RMIN = 1.0E10
      DO 100 I=1,N
      IF(WORK2(I).LT.RMIN) RMIN=WORK2(I)
  100 CONTINUE
      DO 150 I=1,N
      WORK2(I) = WORK2(I) - RMIN
  150 CONTINUE
C
C**   ENCODE AND PLOT TITLE
      ENCODE(36,2000,TEXT(1),ID,ERR=200) NUMPLT,ISTASH
      XT = 1.0
      YT = 11.0
      CALL NEWPN(2)
      CALL SYMBOL(XT,YT,.21,TEXT(1),0.0,36)
      GO TO 201
  200 WRITE(XOTFIL,1000)
  201 CONTINUE
C
      OLDTOP = 0.0
      CALL PLOT(2.0,2.0,3)
      CALL NEWPN(WATPEN)
      CALL EZSRFC(WORK2,IPR,JPR,ANGH(IQ),ANGV,FLAT,EYE,WORK3)
      RETURN
C
 1000 FORMAT(/' TITLE ENCODING ERROR'//)
 2000 FORMAT('TERRAIN:  PLOT # = ',I2,'  TER # = ',I2)
C
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE STRMCN
C
      INCLUDE FILES
      INCLUDE HOWPLT
      INCLUDE MRCHDR
      INCLUDE WORK1
      INCLUDE WORK2
      INCLUDE WORK3
      INCLUDE CONV
C
C**   DETERMINE FILE BLOCK NUMBER FROM FILE RUN NUMBER
      IBLOCK = 3*ISTASH
C
C**   BLOCK IN AND PREP DATA
      CALL DPREP1
      IPR = IPR-1
      JPR = JPR-1
      HXINT = .5*PLH/JPR
      CALL DPREP2(WORK2,IBLOCK)
      IBLOCK = IBLOCK+1
      CALL DPREP2(WORK3,IBLOCK)
C
C**   INITIALIZE PLOTTER AND SET PLOT PARAMETERS
      CALL PLOT(1.0+HXINT,1.0+HXINT,3)
      SPLH = PLH-2.0*HXINT
      PLTHT = -SPLH
      AL = .01*SPLH
      CALL NEWPN(WATPEN)
C
C**   CALL STREAMLINE PLOTTING ROUTINE
      CALL STRMLN(WORK2,WORK3,IPR,JPR,WORK1,WORK4,WORK5,AL,PLTHT)
C
      IF(NCON.EQ.0) GO TO 150
C**   COMPUTE VELOCITY MAGNITUDES
      DO 100 I=1,IPR
      DO 100 J=1,JPR
      INDEX = I + (J-1)*IPR
      WORK1(INDEX) = SQRT(WORK2(INDEX)**2+WORK3(INDEX)**2)
  100 CONTINUE
C
C
C**   INITIALIZE PLOTTER AND SET PLOT PARAMETERS
      CALL PLOT(1.0+HXINT,1.0+HXINT,3)
      NDEC = 1
      PLTHT = SPLH
      NCONT = -NCON
      NSM   = 3
      CALL NEWPN(1)
C
```

```
C**   CALL CONTOUR PLOTTING ROUTINE
      CALL KSOCON(WORK1,WORK2,IPR,JPR,PLTHT,NCONT,CONV,NDEC,NSM)
C
  150 CONTINUE
C**   SET OLDTOP FOR POSSIBLE STACKING
      OLDTOP = PLH + 3.5
C
C**   ENCODE AND PLOT TITLE
      ENCODE(36,3000,TEXT(1),ID,ERR=200) DELTAQ(ISTASH),
     1           TERBLK(ISTASH), ISTASH
      XT = 1.0
      YT = PLH + 2.0
      CALL NEWPN(2)
      IF(OVRLAY.NE.1) CALL SYMBOL(XT,YT,.15,TEXT(1),0.0,36)
      ENCODE(36,2000,TEXT(1),ID,ERR=200) XWIND(ISTASH),YWIND(ISTASH)
      XT = 1.0
      YT = PLH + 2.25
      IF(OVRLAY.NE.1) CALL SYMBOL(XT,YT,.15,TEXT(1),0.0,36)
C
      RETURN
  200 WRITE(XOTFIL,1000)
      RETURN
C
 1000 FORMAT(/' TITLE ENCODING ERROR'//)
 2000 FORMAT('DRIVING WINDS:  VE= ',F5.1,'  VN= ',F5.1)
 3000 FORMAT('BUOY =',F6.1,'  TER # = ',I2,'  WND # = ',I2)
      END




      COMPILER(DIAG=3)
      SUBROUTINE DPREP1
C
      INCLUDE HOWPLT
C
C**   DETERMINE PLOT DIMENSIONS
      IPR  = ILE - ILW +1
      JPR = JLN - JLS +1
      IF(SAMPLE.EQ.1) GO TO 75
      IPR  =  (IPR+1)/SAMPLE
      JPR  =  (JPR+1)/SAMPLE
   75 CONTINUE
C
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE DPREP2(W,IB)
C
      INCLUDE HOWPLT
      INCLUDE WORK1
C
      DIMENSION W(1)
C
C
      CALL BLKIN(NGRD,WORK1(1),IB,WATFIL,IST)
C
      DO 100 I=1,IPR
      DO 100 J=1,JPR
      INDEX1 = (I-1)*SAMPLE + ILW   +  ((J-1)*SAMPLE + JLS - 1)*ILL
      INDEX2 = I + (J-1)*IPR
      W(INDEX2) = WORK1(INDEX1)
  100 CONTINUE
C
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE DPREP3(IP,JP,X,Y,VX,VY,IB,XINT)
C
      DIMENSION X(IP,JP),Y(IP,JP), VX(IP,JP), VY(IP,JP)
C
      CALL DPREP2(VX,IB)
      IB = IB+1
      CALL DPREP2(VY,IB)
C
      VMAX = 0.0
      DO 50 I=1,IP
      DO 50 J=1,JP
      VMAG = VX(I,J)**2 + VY(I,J)**2
      IF(VMAG.GT.VMAX) VMAX=VMAG
   50 CONTINUE
      VMAX = SQRT(VMAX)
C
      DO 100  I=1,IP
      DO 100 J=1,JP
      VX(I,J) = VX(I,J)*XINT/VMAX
      VY(I,J) = VY(I,J)*XINT/VMAX
      X(I,J) = (FLOAT(I)-.5)*XINT - .5*VX(I,J)
      Y(I,J) = (FLOAT(J)-.5)*XINT - .5*VY(I,J)
  100 CONTINUE
C
      RETURN
      END
```

D.6 TERRAIN PROCESSING ROUTINES (TERPRO) OF MRC*UTILITY.

D.6.1 Procedure Elements of Terrain Processor (TERPRO)

```
TBLHDR  PROC
        PARAMETER IDIM=50, THDRSZ=251
        COMMON /TBLHDR/ NNBLK,NDIMXH(IDIM),NDIMYH(IDIM),DELH(IDIM),
       1                SWUTMX(IDIM),SWUTMY(IDIM)
 END
BIGHDR  PROC
        PARAMETER NPNTS=600, BHDRSZ=1241
        COMMON /BIGHDR/ DUM(40),XPNT(NPNTS), YPNT(NPNTS), NDBC
 END
INPUT PROC
        COMMON /INPUT/ UTMX,UTMY,NSQRX,NSQRY,IQUAD
 END
INPEUR PROC
        COMMON /INPUT/ UTMX,UTMY,IDMN,JDMN
 END
FILES  PROC
        COMMON /FILES/ XINFIL,XOTFIL,BIGFIL,MOTFIL
        INTEGER XINFIL,XOTFIL,BIGFIL,MOTFIL
 END
GRID  PROC
        COMMON /GRID/NDIMX,NDIMY,DEL,DISPX,DISPY,NAVRGE
 END
BLKS  PROC
        COMMON /BLKS/ BLKID(9)
        INTEGER BLKID
 END
TERAIN  PROC
        COMMON /TERAIN/ H(40,40)
 END
TERRUF  PROC
        PARAMETER IEUR=51,JEUR=31
        COMMON /TERRUF/ H(IEUR,JEUR), ROUGH(IEUR,JEUR), IDATA(IEUR,JEUR)
 END
```

D.6.2 Subroutines of Terrain Processor (TERPRO)

```
      COMPILER(DIAG=3)
C     PROGRAM BLKDRV
C
      INCLUDE BIGHDR
      INCLUDE TBLHDR
      INCLUDE INPUT
      INCLUDE FILES
C
      NAMELIST /INPUT/ UTMX,UTMY,NSQRX,NSQRY,IQUAD,ISTOP
C
C     ASSIGN FILE LOGICAL UNIT NUMBERS
      XINFIL = 10
      XOTFIL =  6
      BIGFIL =  1
      MOTFIL =  2
      MICTRF = 16
C
C     READ HEADERS INTO COMMON BLOCKS
      CALL BLKIN(BHDRSZ,DUM(1),1,BIGFIL,IST)
      CALL BLKIN(THDRSZ,NNBLK,1,MOTFIL,IST)
      IF(NNBLK.EQ.IDIM) GO TO 10
C
C     READ INPUT DESCRIBING TERRAIN BLOCK TO BE CREATED
    1 ISTOP = 0
      READ (XINFIL,INPUT,END=10)
      IF(ISTOP.EQ.1) GO TO 10
      WRITE(XOTFIL,2000) UTMX,UTMY,NSQRX,NSQRY,IQUAD
C
      CALL BLKMGR
C
      GO TO 1
C
   10 CONTINUE
C     OUTPUT UPDATED HEADER BLOCK
      CALL BLKOUT(THDRSZ,NNBLK,1,MOTFIL,IST)
      WRITE(XOTFIL,1000)
C
 1000 FORMAT(//' MRC TERRAIN FILE GENERATION COMPLETE')
 2000 FORMAT(///' INPUT:'/' UTMX',T30,F8.3/' UTMY',T30,F8.3/
     2         NSQRX',T30,I8/' NSQRY',T30,I8/' IQUAD',T30,I8)
C
C
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE BLKMGR
C
C
      INCLUDE INPUT
      INCLUDE BIGHDR
      INCLUDE FILES
      INCLUDE TBLHDR
      INCLUDE GRID
      INCLUDE TERAIN
      INCLUDE BLKS
C
      DATA HLO/1000./, HDIF/3000./
C
C
      IFLG = 0
      CALL REGRID
      CALL BLOCKO(IFLG)
      IF(IFLG.NE.-1) GO TO 10
      WRITE(XOTFIL,1000)
      RETURN
   10 CONTINUE
      CALL PASTE
C
C     SET HEADER VALUES
      NNBLK = MOD(NNBLK,IDIM) + 1
      NDIMXH(NNBLK) = NDIMX
      NDIMYH(NNBLK) = NDIMY
      DELH(NNBLK) = DEL
      ITMP = BLKID(1)
      SWUTMX(NNBLK) = XPNT(ITMP)+DISPX
      SWUTMY(NNBLK) = YPNT(ITMP)+DISPY
C
      ITER = NNBLK*2
      IRUF = ITER+1
      CALL BLKOUT(1600,H(1,1),ITER,MOTFIL,IST)
      DO 100 I=1,40
      DO 100 J=1,40
      H(I,J) = .4*(H(I,J)-HLO)/HDIF + .1
  100 CONTINUE
      CALL BLKOUT(1600,H(1,1),IRUF,MOTFIL,IST)
C
      WRITE(XOTFIL,2000) NNBLK,NDIMXH(NNBLK),NDIMYH(NNBLK),DELH(NNBLK),
     1                   SWUTMX(NNBLK),SWUTMY(NNBLK)
C
 1000 FORMAT(//' NO BLOCKS GENERATED'//)
 2000 FORMAT(//' RUN NUMBER',T30,I8/
     1         ' X DIMENSION',T30,I8/
     2         ' Y DIMENSION',T30,I8/
     3         ' GRID SPACING',T30,F8.2,' METERS'/
```

```
     4          ' SW UTM X COORDINATE',T30,F8.2,' KM'/
     4          ' SW UTM Y COORDINATE',T30,F8.2,' KM'/)
C
      RETURN
      END




      COMPILER(DIAG=3)
      SUBROUTINE REGRID
C
      INCLUDE INPUT
      INCLUDE GRID
      INCLUDE FILES
C
      DATA STDDEL/63.5/
C
      M = MAX0(NSQRX,NSQRY)
C
      IF(IQUAD.NE.0) GO TO 10
C
      IF(M.EQ.1) NAVRGE=2
      IF(M.EQ.2) NAVRGE=4
      IF(M.EQ.3) NAVRGE=5
      NQ = 80/NAVRGE
      DISP = (NAVRGE-1)*.0005*STDDEL
      NDIMX = NQ*NSQRX
      NDIMY = NQ*NSQRY
      IF (M.EQ.3) NDIMX = 40
      IF (M.EQ.3) NDIMY = 40
      DEL   = STDDEL*NAVRGE
      DISPX = DISP
      DISPY = DISP
      RETURN
C
 10   DISP = STDDEL*.04
      DISPX = 0
      DISPY = 0
      IF(IQUAD.EQ.2.OR.IQUAD.EQ.3) DISPX=DISP
      IF(IQUAD.EQ.3.OR.IQUAD.EQ.4) DISPY=DISP
C
      NDIMX  = 40
      NDIMY   = 40
      DEL    = STDDEL
      NAVRGE = 1
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE BLOCKO(IFLG)
C
      INCLUDE BLKS
      INCLUDE BIGHDR
      INCLUDE INPUT.
      INCLUDE GRID
C
      PARAMETER FFF=39
      DIMENSION IDXSTA(FFF),STADST(FFF),XX(FFF),YY(FFF)
      DIMENSION IDTMP(5),DSTMP(5)
      DATA DSTMAX/14.5/, GRDSIZ/5.08/
C
C     OBTAIN LIST OF GRID POINTS CLOSE TO UTMX,UTMY
      CALL CLOSTN(UTMX,UTMY,XPNT,YPNT,NSTATN,IDXSTA,STADST,NDBC,
     1     DSTMAX)
      IF(NSTATN.LE.0) GO TO 105
C
C
C     SAVE CLOSEST POINTS
      DO 50 I=1,NSTATN
      ITMP = IDXSTA(I)
      XX(I) = XPNT(ITMP)
      YY(I) = YPNT(ITMP)
   50 CONTINUE
C
      GO TO 200
  105 CONTINUE
      IFLG = -1
      RETURN
C
  200 XCORNR = XX(1)
      YCORNR = YY(1)
C
C     DETERMINE INDICES OF BLOCKS DESIRED
      K = 0
      DO 300 I=1,NSQRX
      DO 250 J=1,NSQRY
      X0 = XCORNR + GRDSIZ*(I-1)
      Y0 = YCORNR + GRDSIZ*(J-1)
      CALL CLOSTN(X0,Y0,XX,YY,NSTMP,IDTMP,DSTMP,NSTATN,1.0)
      IF(NSTMP.LT.1) GO TO 105
      L = IDTMP(1)
      K = K+1
      BLKID(K) = IDXSTA(L)
  250 CONTINUE
  300 CONTINUE
C
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE PASTE
C
      INCLUDE TERAIN
      INCLUDE BLKS
      INCLUDE INPUT
      INCLUDE GRID
C
      DIMENSION IDATA(3,3)
      DATA IDATA /40,40,40,20,20,20,16,16,8/
C
      NTMP = 80/NAVRGE
      MAXNSQ = MAX0(NSQRX,NSQRY)
      K = 0
C
      DO 200 IN=1,NSQRX
      DO 100 JN=1,NSQRY
      K = K+1
      NBLK = BLKID(K) + 1
      ILIM = IDATA(IN,MAXNSQ)
      JLIM = IDATA(JN,MAXNSQ)
      IBASE = NTMP*(IN-1)
      JBASE = NTMP*(JN-1)
C
      CALL FILTER(IBASE,JBASE,ILIM,JLIM,NBLK)
  100 CONTINUE
  200 CONTINUE
C
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE FILTER(IBASE,JBASE,ILIM,JLIM,NBLK)
C
      INCLUDE TERAIN
      INCLUDE INPUT
      INCLUDE FILES
      INCLUDE GRID
C
      DIMENSION IBLOCK(80,80)
      CALL BLKIN(6400,IBLOCK(1,1),NBLK,BIGFIL,ISTS)
C
  100 IF(IQUAD.EQ.0) GO TO 200
C
C
      IDISP = 0
      JDISP = 0
      IF(IQUAD.EQ.2.OR.IQUAD.EQ.3) IDISP=40
      IF(IQUAD.EQ.3.OR.IQUAD.EQ.4) JDISP=40
      DO 110 J=1,JLIM
      DO 110 I=1,ILIM
      II = I+IDISP
      JJ = J+JDISP
  110 H(I,J) = IBLOCK(JJ,II)*.304801
      RETURN
C
  200 N = NAVRGE
      DO 250 J=1,JLIM
      DO 250 I=1,ILIM
      I1 = N*I+1
      J1 = N*J+1
      AV = 0.0
        DO 205 K=1,N
        DO 205 L=1,N
        I2 = I1-K
        J2 = J1-L
        AV = AV+IBLOCK(I2,J2)
  205   CONTINUE
      AV = AV/(N*N)
      ICOUNT = 0
      SUM = 0
        DO 210 K=1,N
        DO 210 L=1,N
        I2 = I1-K
        J2 = J1-L
        DIFF = IBLOCK(I2,J2)-AV
        IF(DIFF.GT.0.0) ICOUNT=ICOUNT+1
        SUM = SUM+DIFF**2
  210   CONTINUE
      SD = SQRT(SUM/(N*N-1))
      AVHT = AV
```

```
      NTESTH = (3*N*N)/4
      NTESTL = (N*N)/4
      IF(ICOUNT.GE.NTESTH) AVHT=AVHT+SD
      IF(ICOUNT.LE.NTESTL) AVHT=AVHT-SD
      JM = J+JBASE
      IM = I+IBASE
      H(JM,IM) = AVHT*.304801
  250 CONTINUE
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE CLOSTN(XIN,YIN,X,Y,NSTATN,
     1      IDXSTA,STADST,STNMAX,DISTMN)
C
C
C
      DIMENSION X(1),Y(1), IDXSTA(1),STADST(1)
C
      INTEGER STNMAX
C
C     DETERMINE THE CLOSEST STATION WITHIN DISTANCE DISTMN AND ORDER
C     STATIONS IN ASCENDING ORDER OF DISTANCE
C
      NSTATN = 0
      XSOURS = XIN
      YSOURS = YIN
      N = STNMAX
      DO 100 I=1,N
C     CALCULATE DISTANCE FROM RELEASE TO STATION
      D = SQRT((X(I)-XSOURS)**2+(Y(I)-YSOURS)**2)
      IF (D .GT. DISTMN) GO TO 100
      NSTATN = NSTATN +1
      IDXSTA(NSTATN) = I
      STADST(NSTATN) = D
  100 CONTINUE
      IF (NSTATN .LT. 2 ) GO TO 120
      DO 110 J=2,NSTATN
      DO 110 I=2,NSTATN
      IF (STADST(I-1) .LE. STADST(I)) GO TO 110
      D = STADST(I)
      K = IDXSTA(I)
      STADST(I) = STADST(I-1)
      IDXSTA(I) = IDXSTA(I-1)
      STADST(I-1) = D
      IDXSTA(I-1) = K
  110 CONTINUE
  120 RETURN
      END
```

```
C     PROGRAM BLKDRV
C
      INCLUDE TBLHDR
      INCLUDE INPEUR
      INCLUDE FILES
C
      NAMELIST /INPUT/ UTMX,UTMY,IDMN,JDMN,ISTOP
C
C     ASSIGN FILE LOGICAL UNIT NUMBERS
      XINFIL = 10
      XOTFIL =  6
      MOTFIL =  2
      MICTRF = 16
C
C
C     READ HEADER INTO COMMON BLOCK
      CALL BLKIN(THDRSZ,NNBLK,1,MOTFIL,IST)
      IF(NNBLK.EQ.IDIM) GO TO 10
C
    5 CONTINUE
C**   INITIALIZE*
      ISTOP = 0
      IDMN = 51
      JDMN = 31
C     READ INPUT DESCRIBING TERRAIN BLOCK TO BE CREATED
      READ (XINFIL,INPUT,END=10)
      IF(ISTOP.EQ.1) GO TO 10
      WRITE(XOTFIL,2000) UTMX,UTMY,IDMN,JDMN
C
      CALL BLKMGR
C
      GO TO 5
C
   10 CONTINUE
C     OUTPUT UPDATED HEADER BLOCK
      CALL BLKOUT(THDRSZ,NNBLK,1,MOTFIL,IST)
      WRITE(XOTFIL,1000)
 1000 FORMAT(//' MRC EURO TERRAIN FILE GENERATION COMPLETE')
 2000 FORMAT(///' INPUT:'/' UTMX',T30,F8.3/' UTMY',T30,F8.3/
     2       ' IDMN',T30,I8/' JDMN',T30,I8)
C
      END
```

```
      SUBROUTINE BLKMGR
C
C
      INCLUDE INPEUR
      INCLUDE FILES
      INCLUDE TBLHDR
      INCLUDE TERRUF
      PARAMETER IOUT = IEUR*JEUR
C
      DATA DEL/100.0/
C
C
      CALL CORNER(IS,JS,XPNT,YPNT)
      IF(IS.LT.0) GO TO 50
      IF(JS.LT.0) GO TO 50
      IF(IS+IDMN.GT.101) GO TO 50
      IF(JS+JDMN.GT.51) GO TO 50
      GO TO 100
   50 CONTINUE
      WRITE(XOTFIL,1000)
      RETURN
  100 CONTINUE
      CALL CARVE(IS,JS,IDATA,IDMN,JDMN)
      NGRD = IDMN*JDMN
      CALL UNPACK(IDATA,H,ROUGH,NGRD)
C
C     SET HEADER VALUES
      NNBLK = MOD(NNBLK,IDIM) + 1
      NDIMXH(NNBLK) = IDMN
      NDIMYH(NNBLK) = JDMN
      DELH(NNBLK) = DEL
      SWUTMX(NNBLK) = XPNT
      SWUTMY(NNBLK) = YPNT
C
      ITER = NNBLK*2
      IRUF = ITER+1
      CALL BLKOUT(IOUT,H(1,1),ITER,MOTFIL,IST)
      CALL BLKOUT(IOUT,ROUGH(1,1),IRUF,MOTFIL,IST)
C
      WRITE(XOTFIL,2000) NNBLK,NDIMXH(NNBLK),NDIMYH(NNBLK),DELH(NNBLK),
     1                   SWUTMX(NNBLK),SWUTMY(NNBLK)
C
 1000 FORMAT(//' NO BLOCKS GENERATED'//)
 2000 FORMAT(//' RUN NUMBER',T30,I8/
     1         ' X DIMENSION',T30,I8/
     2         ' Y DIMENSION',T30,I8/
     3         ' GRID SPACING',T30,F8.2,' METERS'/
     4         ' SW UTM X COORDINATE',T30,F8.2,' KM'/
     4         ' SW UTM Y COORDINATE',T30,F8.2,' KM'/)
C
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE CORNER(IS,JS,XPNT,YPNT)
C
      INCLUDE INPEUR
C
      DATA DEL/0.1/, XCORN/552.4/, YCORN/5592.665/, ANG/.89982/
C
      C1 = COS(ANG)
      C2 = SIN(ANG)
C
      XTMP = (UTMX-XCORN)/DEL
      YTMP = (UTMY-YCORN)/DEL
      IS = XTMP*C1+YTMP*C2 + .01
      JS = -XTMP*C2+YTMP*C1 + .01
C
      XPNT = XCORN + DEL*(IS*C1-JS*C2)
      YPNT = YCORN + DEL*(IS*C2+JS*C1)
C
      RETURN
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE CARVE(IS,JS,IDATA,IDIM,JDIM)
      COMMON/MAPCOM/E,NY,XMAX,YMAX,ZMAX,M(51,101)
      DIMENSION IDATA(IDIM,JDIM)
      DO 8 J = 1,JDIM
      DO 6 I = 1,IDIM
      IDATA(I,J) = M(JS+J,IS+I)
    6 CONTINUE
    8 CONTINUE
   10 CONTINUE
C
      END
```

```
      COMPILER(DIAG=3)
      SUBROUTINE UNPACK(IDATA,H,ROUGH,NGRD)
C
      DIMENSION H(NGRD),ROUGH(NGRD),IDATA(NGRD), VEG(7)
      DATA VEG /.099,2.999,9.999,9.999,15.001,7.499,7.499/
C
      DO 10 I = 1,NGRD
      K = IDATA(I) / 100000
      L = K / 10
      K = K - 10 * L
      H(I) = L+VEG(K+1)
      ROUGH(I) = VEG(K+1)*.1
   10 CONTINUE
C
      RETURN
      END
```

D. SAMPLE INPUT ELEMENTS FOR PLOTTING (PLTPRG) AND TERRAIN PROCESSING (TERPRO) ROUTINES OF MRC*UTILITY

```
***   SAMPLE INPUT ELEMENTS FOR TERRAIN AND PLOTTING ROUTINES
***   MT.TERINP
 $INPUT  UTMX=351.3, UTMY=3585.7, NSQRX=1, NSQRY=1, IQUAD=0, ISTOP=0
 $END
 $INPUT  UTMX=351.3, UTMY=3585.7, NSQRX=1, NSQRY=1, IQUAD=4, ISTOP=0
 $END
 $INPUT  UTMX=345.5, UTMY=3585.8, NSQRX=2, NSQRY=2, IQUAD=0, ISTOP=0
 $END
 $INPUT  UTMX=345.5, UTMY=3585.8, NSQRX=3, NSQRY=3, IQUAD=0, ISTOP=0
 $END
 $INPUT  UTMX=346.4, UTMY=3580.8, NSQRX=3, NSQRY=3, IQUAD=0, ISTOP=0
 $END
 $INPUT  UTMX=331.2, UTMY=3591.2, NSQRX=1, NSQRY=1, IQUAD=4, ISTOP=0
 $END
 $INPUT  UTMX=351.1, UTMY=3570.5, NSQRX=2, NSQRY=2, IQUAD=0, ISTOP=0
 $END
 $INPUT  UTMX=356.7, UTMY=3595.8, NSQRX=2, NSQRY=2, IQUAD=0, ISTOP=0
 $END
 $INPUT  UTMX=356.7, UTMY=3595.8, NSQRX=1, NSQRY=1, IQUAD=0, ISTOP=0
 $END
 $INPUT  UTMX=352.0, UTMY=3616.2, NSQRX=2, NSQRY=2, IQUAD=0, ISTOP=0
 $END
***   MT.TERINP/EUR
 $INPUT  UTMX=552.7, UTMY=5593.3
 $END
***   MT.PLTINP
 $END
 $INPLOT  ISTASH=1,  ILL=40,  JLL=40,
          WATFIL=17,  ILE=40,  ILW=1,  JLN=40,  JLS=1,
          ITYPE=4,  WATPEN=3,  OVRLAY=0,
          NUMUP=2,  PLH=14.0,  SAMPLE=2,  NCON=8,
          ISTOP=0
 $END
```

☆ U.S. GOVERNMENT PRINTING OFFICE: 1977 777-093/39